Dissertation

From Local Visual Homing Towards Navigation of Autonomous Cleaning Robots

Der Technischen Fakultät
der Universität Bielefeld
vorgelegt von

Lorenz Hillen

zur Erlangung des Grades eines
Doktors der Ingenieurwissenschaften.

2013

Eröffnung des Promotionsverfahrens:	24. April 2013
Tag der Verteidigung:	28. August 2013
Erstgutachter:	Prof. Dr. Ralf Möller
Zweitgutachter:	Jun.-Prof. Dr. Vanessa Hafner
Drittgutachter:	Assoc. Prof. Andrew Vardy
Vorsitzende der Prüfungskommission:	Prof. Dr. Barbara Hammer
Mittelbauvertreter:	Dr. Carsten Gnörlich

Für meinen Vater
(To my father)

Abstract

This thesis deals with the implementation of navigation strategies for a domestic floor-cleaning robot operating on omnidirectional images as primary sensory information. Such navigation strategies enable the robot to efficiently cover its entire workspace while avoiding both uncleaned areas and repeated coverage. This is accomplished (i) by systematically guiding the robot along meandering lanes, i.e. along straight lanes placed next to each other at a predefined and constant distance and (ii) by building a map of the robot's environment to distinguish cleaned and uncleaned areas. Since domestic cleaning robots are considered consumer goods, they can only be equipped with a limited number of cheap sensors and restricted computational power. This fact poses additional challenges onto the design of navigation strategies for domestic floor-cleaning robots.

The navigation strategies described in this thesis use omnidirectional images, in our case panoramic images with a full 360° horizontal field of view. We consider omnidirectional cameras an appropriate choice because they (i) are relatively cheap sensors, (ii) provide dense sensory information about the robot's environment, and (iii) are multi-purpose sensors applicable to further aspects of cleaning-robot navigation beyond the scope of this thesis (e.g. obstacle detection, visual odometry, or user interaction). We characterize a position in space by the entire omnidirectional image acquired at this place (hence the methods belong to the class of appearance-based navigation methods) without detecting visible features in the image. Several places are integrated into a dense topo-metric map of the robot's environment. Such maps (i) offer a metrical position estimate required for guiding the robot along meandering lanes, (ii) have a spatial resolution which is fine enough for accurate navigation, (iii) can be easily built from the available sensor data, and (iv) allow for efficiently operating on the maps. Spatial relations between places stored in the map are estimated by applying a local visual homing method. Such methods are parsimonious yet robust and accurate methods for partial ego-motion estimation from visual information. They recover the direction (but not the distance) of the translation and the rotation of the robot's motion between two images acquired in direct vicinity of each other without physically moving between places. As far as we know, our navigation methods are the first application of omnidirectional vision, dense topo-metric maps, and local visual homing for the control of cleaning robots. Hence, this thesis is also a feasibility study to prove the applicability of these concepts for navigation of cleaning robots. Since a complete control scheme for a cleaning robot is beyond the scope of this thesis, we propose two essential substrategies of such a control scheme: (i) vision-based trajectory control and mapping and (ii) visual detection of already cleaned areas.

Regarding *trajectory control and mapping*, we propose a mostly vision-based controller for covering a rectangular area of the entire workspace by meandering lanes. While moving along a lane (and cleaning), the robot adds snapshots at regular distances to its dense topo-metric map, which are used on the subsequent lane to estimate the robot's current distance to the previous lane. For this purpose, the bearing from the current position towards at least two snapshots stored along the previous lane is taken by applying local visual homing. The bearing information and an odometry-based estimate of the distance between the two considered snapshots are fused in order to estimate the robot's current distance to the previous lane. The robot is kept on a lane parallel to the previous one by keeping the distance to the previous lane at a predefined value. Instead of estimating the robot's full pose as performed by common navigation strategies, we only estimate the distance to the previous lane and the robot's current orientation to avoid unnecessary computations. The results obtained from real-robot experiments reveal that the algorithm is capable of guiding the robot along parallel and meandering cleaning lanes with only a small portion of gaps or overlap between lanes.

Detecting already cleaned areas is essential in order to avoid repeated coverage or uncleaned areas between neighboring cleaning segments. This detection is a special instance of the loop-closure detection problem usually occurring during map-building whenever the robot approaches an already mapped area. We solve the loop-closure problem by two different approaches both incorporating pairwise image comparisons between the robot's currently perceived image and several images stored in the map. The first approach, referred to as *holistic approach*, relies on pixel-by-pixel comparisons of the considered images. The second, referred to as *signature-based approach*, computes low-dimensional signatures extracted from the entire image and compares these instead of the images themselves.

Pixel-based approaches require the application of a compass to align images prior to image comparison. The standard compass method rotates one of the images step-by-step while keeping the other fixed and repeatedly compares the images to search for the best match. We propose an accelerated variant of this method operating in the Fourier domain. This method is capable of computing the best match without repeatedly shifting and comparing images. In order to achieve robustness against illumination changes, we preprocess images prior to comparison and apply illumination-tolerant comparison functions. Loop-closure detection and compass accuracy were assessed by image-database experiments systematically evaluating a wide range of different preprocessing and comparison techniques. Regarding loop-closure detection, holistic methods achieve very good detection results even for strong illumination changes. The proposed Fourier-based compass is more efficient than the standard method, but does not achieve its accuracy. Due to their computational complexity, the tested holistic approaches to loop-closure detection are —at least with the current implementation— not suitable for a real-robot application.

Signature-based approaches allow for efficient image comparisons because they rely on low-dimensional and rotation-invariant image descriptors. Due to their rotational invariance, signatures can be compared without prior compass alignment as required by holistic methods. To measure the accuracy of loop-closure detection, we performed image-database experiments which systematically tested different combinations of signatures and comparison functions operating on the images' intensity information without prior preprocessing. The tested methods allow for accurate loop-closure detection under constant illumination. However, detection is likely to fail under moderate or strong illumination changes. For the most promising combination of signature and comparison function, real-robot experiments were conducted leading to similar results. These results are surprising because we tested several combinations of signatures and comparison functions which should theoretically tolerate illumination changes better than the combination performing best in our experiments. Despite their low tolerance against illumination changes, which need to be increased in future work, we favor the application of signature-based approaches because of their low computational complexity.

The overall results of this thesis clearly reveal that omnidirectional vision, dense topo-metric maps, and local visual homing are appropriate building blocks for visual control of cleaning robots because they allow for efficient, accurate, and robust navigation. We conclude that dense topo-metric maps are suitable representations of space —both for trajectory control and for detecting already cleaned areas. Thus, the navigation strategies proposed in this thesis can be used as a basis for a more complex control architecture enabling the robot to completely cover complex-shaped areas. This includes mechanisms to detect and approach uncleaned areas based on map information and to combine several segments of meandering lanes as obtained from our trajectory controller. Using omnidirectional vision not only for navigation but also for obstacle detection, odometry, or user interaction could be a promising means to reduce hardware costs of a potential product by avoiding dedicated sensors.

Zusammenfassung

Diese Dissertation beschreibt die Implementation von Navigationsstrategien für einen mobilen Bodenreinigungsroboter für den Haushaltsgebrauch, der omnidirektionale Bilder als primäre Sensorinformation verwendet. Die Navigationsstrategien ermöglichen die vollständige und effiziente Reinigung der gesamten dem Roboter zugänglichen Fläche unter gleichzeitiger Vermeidung von nicht oder mehrfach befahrenen Flächen. Die Anforderungen lassen sich durch die Wahl einer geeigneten Fahrstrategie und den Aufbau einer Karte der Umgebung erfüllen. Als Fahrstrategie zur systematischen Überdeckung des Raumes werden dazu häufig mäandrierende Bahnen gewählt, also gerade Bahnen, die in einem vorgegebenen, konstanten Abstand zueinander verlaufen. Die Karte ermöglicht eine Unterscheidung zwischen bereits befahrenen und noch nicht befahrenen Gebieten. Reinigungsroboter für den Haushaltsgebrauch sind Konsumgüterartikel. Sie können folglich nur mit einer begrenzen Anzahl an kostengünstigen Sensoren und mit kostengünstiger und daher leistungsschwacher Hardware ausgestattet werden. Dadurch entstehen zusätzliche Herausforderungen für die Umsetzung von Reinigungsstrategien.

Die in dieser Arbeit vorgestellten Navigationsstrategien für Bodenreinigungsroboter verwenden als primäre Sensorinformation omnidirektionale Bilder. In unserem Fall handelt es sich dabei um Panoramabilder mit einem horizontalen Sichtbereich von 360°. Wir betrachten omnidirektionale Kameras als geeignete Wahl, weil es sich dabei um relativ günstige Sensoren handelt und weil sie dichte Sensorinformation über die Umwelt des Roboters liefern. Über die im Rahmen dieser Arbeit betrachteten Aspekte hinaus können solche Kameras beispielsweise auch zur Hinderniserkennung, für visuelle Odometrieberechnungen oder zur Benutzerinteraktion angewendet werden. In dieser Arbeit werden Orte durch Panoramabilder charakterisiert wie sie am jeweiligen Ort aufgenommen wurden. Bildinformationen von mehreren Orten werden in eine dichte topo-metrische Karte integriert, welche die Umgebung des Roboters repräsentiert. Für unseren Anwendungsfall bieten diese Karten die folgenden vier Vorteile: (i) sie speichern metrische Positionsinformation, die benötigt wird, um parallele Bahnen abzufahren, (ii) sie haben eine ausreichend feine räumliche Auflösung, die eine präzise Navigation des Roboters ermöglicht, (iii) sie können einfach aus der zur Verfügung stehenden visuellen Information aufgebaut werden, und (iv) die Nutzung solcher Karten ist ohne großen Rechenaufwand möglich. Räumliche Beziehungen zwischen gespeicherten Orten werden durch Verfahren zur „visuellen Zielanfahrt“[1] geschätzt. Solche Verfahren sind effiziente und trotzdem genaue und zuverlässige Algorithmen zur partiellen Eigenbewegungsschätzung aus visueller Information. Aus zwei Bildern, die an zwei nahe beieinander liegenden Orten aufgenommen wurden, werden die Richtung der Translationskomponente (nicht jedoch deren absolute Länge) und die Rotationskomponente der dazwischenliegenden Roboterbewegung geschätzt (für die Schätzung muss keine Zielanfahrt erfolgen). Nach unserem Kenntnisstand sind die von uns entwickelten Navigationsstrategien die erste Anwendung dieser Konzepte zur Navigation von mobilen autonomen Reinigungsrobotern für den Haushaltsgebrauch. Die vorliegende Arbeit stellt daher auch eine Machbarkeitsstudie dar, welche die Anwendbarkeit dieser Konzepte zur Navigation von Reinigungsrobotern zeigen soll. Die Arbeit beschränkt sich auf zwei wesentliche Bestandteile einer Kontrollarchitektur für einen Reinigungsroboter: (i) visuelle Bahnsteuerung mit gleichzeitigem Kartenaufbau und (ii) visuelle Erkennung bereits befahrener Gebiete.

[1] Üblicherweise wird zur Bezeichnung dieser Verfahren auch im Deutschen der englische Begriff „*local visual homing*“ verwendet.

Zur *visuellen Bahnsteuerung mit gleichzeitigem Kartenaufbau* schlagen wir einen Regelungsalgorithmus vor, der es ermöglicht ein einzelnes Reinigungssegment mit mäandrierenden Bahnen zu überdecken. Als Reinigungssegment wird einen rechteckiger Bereich der Umgebung bezeichnet; eine vollständige Überdeckung der Umgebung kann durch aneinandersetzen mehrerer Reinigungssegmente erfolgen. Während sich der Roboter auf einer Reinigungsbahn bewegt, werden in regelmäßigen Abständen Kamerabilder aufgenommen und zur Karte hinzugefügt. Diese Bilder werden auf der nachfolgenden Bahn verwendet, um den Abstand des Roboters zur Vorgängerbahn zu schätzen. Dazu werden von der aktuellen Roboterposition aus mindestens zwei Kamerabilder, die entlang der Vorgängerbahn gespeichert wurden, angepeilt. Die so berechnete Richtungsinformation und eine odometriebasierte Abstandsschätzung zwischen den Kamerabildern werden fusioniert, um so den aktuellen Abstand des Roboters zur Vorgängerbahn zu schätzen. Durch Regelung dieses Abstandes auf einen vorgegebenen und konstanten Wert kann der Roboter entlang einer Bahn gesteuert werden, die parallel zu ihrer Vorgängerbahn ist. Die Mehrzahl der in der Literatur vorgeschlagenen Navigationsalgorithmen schätzt die vollständige Pose (also Position und Orientierung im Raum) des Roboters. Im Gegensatz zu diesen Ansätzen, werden durch unser Verfahren nur der Abstand zur Vorgängerbahn und die Orientierung des Roboters geschätzt. Dies vermeidet die Berechnung nicht erforderlicher Information. Das Verfahren wurde in Roboterexperimenten getestet, deren Ergebnisse zeigen, dass das Verfahren geeignet ist, um eine Fläche mit mäandrierenden Bahnen so zu überdecken. Dabei entsteht nur ein geringer Anteil an Lücken oder mehrfach befahrener Fläche.

Die *visuelle Erkennung bereits befahrener Gebiete*[2] ist von Bedeutung, weil dadurch Mehrfachbefahrung und nicht befahrene Gebiete zwischen benachbarten Reinigungssegmenten vermieden werden können. Zur visuellen Erkennung bereits befahrener Gebiete werden zwei unterschiedliche Verfahren betrachtet, die beide auf einem Vergleich des aktuellen Kamerabildes mit mehreren in der Karte gespeicherten Bildern beruhen. Der erste Ansatz wird als *holistischer Ansatz* bezeichnet und vergleicht Bilder Pixel für Pixel. Der zweite Ansatz wird als *signaturbasierter Ansatz* bezeichnet und vergleicht anstelle der Bilder niedrigdimensionale Bildsignaturen, die global aus dem gesamten Bild berechnet werden.

Pixelbasierte Ansätze benötigen ein Kompassverfahren, das die Bilder an einer gemeinsamen Referenzrichtung ausrichtet, bevor diese verglichen werden können. Das Standardverfahren hierzu rotiert eines der Bilder schrittweise, während das zweite Bild konstant gehalten wird. Durch Vergleiche der rotierten Bilder mit dem zweiten Bild wird die beste Übereinstimmung berechnet. Wir schlagen ein Kompassverfahren vor, das im Fourierraum arbeitet und das die beste Übereinstimmung ohne schrittweise Rotation und mit nur einem Bildvergleich berechnen kann. Um Robustheit gegen Beleuchtungsänderungen zu erlangen, wenden wir Bildvorverarbeitungsmethoden und beleuchtungstolerante Bilddistanzfunktionen an. Die Genauigkeit der vorgeschlagenen Verfahren wurde durch Experimente mit Bilddatenbanken ermittelt, in denen systematisch eine Vielzahl an verschiedenen Vorverarbeitungsmethoden und Bilddistanzfunktionen verglichen wurde. Zur Erkennung bereits befahrener Bereiche erzielen die vorgeschlagenen Verfahren sogar unter extremen Beleuchtungsänderungen sehr gute Erkennungsraten. Das vorgeschlagene Kompassverfahren kann effizienter berechnet werden als das Standardverfahren; es erzielt aber nicht dessen Genauigkeit. Aufgrund des Rechenaufwandes sind die vorgeschlagenen Verfahren —zumindest in der derzeit verwendeten Implementierung— nicht für einen Echtzeiteinsatz auf einem Roboter geeignet.

Signaturbasierte Ansätze ermöglichen effiziente Bildvergleiche, weil sie auf niedrigdimensionalen und rotationsinvarianten Bilddeskriptoren beruhen. Auf Grund ihrer Rotationsinvarianz benötigen signaturbasierte Ansätze im Gegensatz zu holistischen Ansätzen keinen Kompass. Um die Erkennungsgenauigkeit von signaturbasierten Ansätzen zu bestimmen wurden Bilddatenbankexperimente durchgeführt, die systematisch verschiedene Kombinationen aus Signaturen und Distanzfunktionen vergleichen. Die Signaturen werden dabei auf den unvorverarbeiteten Intensitätsbildern berechnet.

[2] Im Deutschen wird auch oft der Begriff „*Loopclosure*-Erkennung“ verwendet.

Unter konstanten Beleuchtungsbedingungen erreichen die getesteten Verfahren sehr zuverlässige Erkennungsraten; moderate oder starke Beleuchtungsänderungen führen allerdings zu einer großen Anzahl an Fehlklassifikationen. Diese Ergebnisse sind überraschend, weil eine Reihe an Kombinationen aus Signaturen und Distanzfunktionen getestet wurde, die in der Theorie deutlich robuster sein sollten als die Kombination, die in unseren Experimenten am besten abgeschnitten hat. Trotz ihrer geringen Toleranz gegen Beleuchtungsänderungen, die Nachbesserungen erfordert, bevorzugen wir für weiterführende Arbeiten signaturbasierte Ansätze. Der Hauptgrund hierfür liegt in ihrer effizienten Berechenbarkeit.

Die Gesamtergebnisse dieser Arbeit zeigen, dass omnidirektionale Bildinformation, dichte topometrische Karten und visuelle Zielanfahrt geeignete Grundbausteine für die Entwicklung von visuell gesteuerten Reinigungsrobotern sind. Diese Komponenten ermöglichen die Entwicklung effizienter, genauer und zuverlässiger Navigationsstrategien. Darüber hinaus schließen wir, dass dichte topo-metrische Karten eine geeignete Repräsentation für die visuelle Bahnregelung und die visuelle Erkennung bereits befahrener Gebiete sind. Aus diesem Grund können die im Rahmen dieser Arbeit entwickelten Navigationsalgorithmen als Grundlage für weiterführende Navigationstrategien dienen, die es dem Roboter ermöglichen komplexe Umgebungen wie Wohnräume vollständig zu befahren. Solch komplexere Strategien sind beispielsweise die Erkennung und das Anfahren noch nicht gereinigter Gebiete und die Kombination verschiedener Segmente aus mäandrierenden Bahnen. Die Verwendung omnidirektionaler Bildinformation nicht nur zur Navigation sondern auch zur Hinderniserkennung, Odometrieberechnung oder zur Nutzerinteraktion erscheint uns ein vielversprechender Ansatz, um die Hardwarekosten eines möglichen Produktes dadurch zu senken, dass dedizierte Sensoren für diese Anwendungen eingespart werden können.

Die Arbeit ist folgendermaßen gegliedert. Kapitel 1 motiviert diese Arbeit, beschreibt die Zielsetzung und stellt die Gliederung der Arbeit sowie die wissenschaftlichen Beiträge jedes einzelnen Kapitels vor. Daran schließt sich ein Einleitungsteil an, der relevante Aspekte aus den Gebieten der Reinigungsrobotik (Kapitel 2) und der visuellen Navigation mit omnidirektionalen Bildern (Kapitel 3) vorstellt. In den Kapiteln 4 bis 6 werden die entwickelten Navigationsstrategien beschrieben: visuelle Bahnsteuerung mit gleichzeitigem Kartenaufbau (Kapitel 4), holistische Erkennung bereits befahrener Gebiete und Kompassverfahren (Kapitel 5) und signaturbasierte Erkennung bereits befahrener Gebiete (Kapitel 6). Daran schließt sich in Kapitel 7 eine abschließende Diskussion sowie ein Ausblick auf weiterführende Arbeiten an. In den Anhängen A bis D werden weiterführende Daten und mathematische Herleitungen zusammengestellt.

Acknowledgements

Sometimes, when you've a very long street ahead of you, you think how terribly long it is and feel sure you'll never get it swept. [. . .]

You must only concentrate on the next step, the next breath, the next stroke of the broom, and the next, and the next. Nothing else. [. . .] That way you enjoy your work, which is important, because then you make a good job of it. And that's how it ought to be.[. . .]

And all at once, before you know it, you find you've swept the whole street clean, bit by bit.

(Michael Ende in "Momo"; [158] pp. 35–36)

In the above quote, street-cleaner Beppo explains how to clean a seemingly endless street. His principle also holds for finishing PhD projects and for domestic floor-cleaning robots. How cleaning robots can step by step, lane by lane, and segment by segment cover complex workspaces is addressed in the main part of this thesis. Here, I would like to cordially thank the persons and organizations which supported me throughout every single step of the work leading to this dissertation.

First of all, I would like to thank my supervisor Prof. Dr. Ralf Möller (Computer Engineering Group, Faculty of Technology, University Bielefeld) for fruitful discussions, for valuable feedback, and for an infinite number of open research questions including ideas for answering them. He provided me the opportunity to work in an interesting and challenging research field, which combines basic academic research with a strong focus on application-oriented aspects. I sincerely appreciate that he always had an open ear for both scientific questions or non-scientific concerns and always contributed to finding and implementing good solutions.

I am grateful to Jun.-Prof. Dr. Vanessa V. Hafner (Cognitive Robotics Group, Department of Computer Science, Humboldt University of Berlin) and Assoc.-Prof. Dr. Andrew Vardy (joint with Department of Computer Science, Faculty of Science and with Electrical and Computer Engineering, Faculty of Engineering and Applied Science, Memorial University of Newfoundland, St. John's, Canada) for reviewing this thesis and to Prof. Dr. Barbara Hammer (Theoretical Computer Science, Faculty of Technology, University Bielefeld) and Dr. Carsten Gnörlich (Computer Center, Faculty of Technology, Bielefeld University) for joining the thesis committee.

During my time as research associate with the Bielefeld Computer Engineering Group, I received funding from the German Research Foundation: first under the project *"Visual Homing Using Adaptive Optic Flow Algorithms"* (grant number MO 1037/2-1), later from the center of excellence *"Cognitive Interaction Technologies"* (grant number EXC277).

I would like to thank the entire team of the Computer Engineering Group, Birthe Babies, Angelika Deister, Dario Differt, David Fleer, Michael Horst, Alexander Kaiser, Dr. Tim Köhler, Martin Krzykawski, Klaus Kulitza, Dr. Frank Röben, Dr. Wolfram Schenck, and Constanze Schwan, for discussions, help, advice, support, computers operating reliably, assistance with robot-hardware problems, and everything that made life here a pleasant one. In particular, I would like to thank Frank Röben and Martin Krzykawski for their close collaboration during the early phase of the cleaning project.

Furthermore, I would like to thank Jochen Spickerhof (Knowledge-Based Systems Research Group, Institute for Computer Science, Osnabrück) for the insightful discussion on graph-based SLAM section 3.3.2.1. I'm thankful to Dr. Wolfgang Stürzl (now with the German Aerospace Center, Institute of Robotics and Mechatronics, Wessling; formerly with the Neurobiology Group, Faculty of Biology, University Bielefeld) for various coffee breaks, discussions, co-supervizing student projects, and numerous travel tips regarding private holiday destinations. Ten years ago, Wolfgang was my advisor when I was a student assistant at the Tübingen Department of Cognitive Neuroscience, and it was great to meet him again in Bielefeld.

I am grateful to Kristin Belzer, Oliver Schlüter, and Martin Krzykawski for proof-reading this thesis. Martin is the most passionate typo hunter I am aware of and is specialized on rare creatures such as "dessert ants" (for the experts: *Melophorus typoi*) and considerably reduced the number of mistakes in this and many of my other works. I would like to thank Andrew Vardy for patiently answering my questions regarding the English language. Since I started working at the Computer Engineering Group, he is my favorite address for all language issues.

I am particularly grateful to Prof. Dr. José Santos-Victor and Asst.-Prof. Dr. Alexandre Bernardino (both with the Computer Vision Laboratory, Institute for Systems and Robotics, Instituto Superior Técnico, Lisbon) for two lunches during my last visit at Lisbon. At this time, this thesis was in an almost unreachable distance, and I do not know what would have happened without their encouragement and belief in me.

I always enjoyed supervising student projects and getting in contact with students, teaching them, discussing with them, and also learning from them. For this reason I would like to mention (in alphabetical order) Dr. Matthias Behnisch, Björn Böttcher, Johann Engelbrecht, Gereon Götze, Danny Hartwig, Martin Höner, Janina de Jong, Sven Kanies, Marcus Kesting, Dr. Sven Kreft, Andre Lemme, Ioannis Moutogiorgos, Christian Munier, Sebastian Ruwisch, Oliver Schlüter, Thomas Schöpping, Andreas Stöckel, Claudius Strub, and Daniel Venjakob. For me it was a great experience and pleasure to accompany the courses of their bachelor, master, diploma or student projects.

Finally, but most important, I would like to express my honest gratitude to my personal environment for almost endless support. Special thanks to my wife Nina for her patience and for her excellent catering during the last weeks. With her consultant view on problems, she often helped me to structure my thoughts and to see open problems from a different perspective. I would like to thank my mother, Rosi, for her ongoing interest in the course of this work and for her patience in not seeing me too often during the last months of preparing this dissertation. I'm grateful to all the friends in Bielefeld who made me feel at home here. I will for sure remember the numerous lunch breaks, sport sessions, or choir rehearsals with them. Together with my friends from outside Bielefeld, they reminded me that there is a life besides the fascinating field of cleaning-robot navigation.

Contents

List of Figures

Figures requiring color printing are marked by a (†); colored figures, for which color information is not essential, are marked by a ($*$). In the captions of these figures, we state "Figure requires color printing." and "Figure best viewed in color.", respectively. All other (unmarked) figures are grayscale figures.

List of Tables

1. Introduction

In this chapter, we motivate our research on domestic floor-cleaning robots (section 1.1) and introduce relevant building blocks for the proposed navigation strategies (section 1.2). The objectives are presented in section 1.3, and the organization of the thesis is described in section 1.4.

1.1. Motivation

Robots are supposed to facilitate humans' lives by assisting them in tedious, monotonous, physically demanding, or potentially dangerous tasks. While industrial robots are indispensable tools applied in various domains [270, 491], mobile robots have not yet made it into our everyday life [157, 209, 212, 520]. Most of the available mobile robot platforms are research prototypes, and the only robots produced in larger quantities are specialized and expensive machines for small markets like agriculture [43, 152] or underwater robots [11]. The only two exceptions are robotic lawn mowers and floor-cleaning robots for household usage [157, 518, 520]. Floor-cleaning robots became available on the consumer market since the year 2000. Although currently being far from omnipresent, market studies predict a large and even growing market potential for domestic floor-cleaning robots ([157, 209, 212, I71, 518, 520], detailed market studies: [300, 695]).

Against this background, we decided to apply our experience in vision-based robot navigation to a new research domain: navigation strategies for a domestic floor-cleaning robot relying on omnidirectional vision. The exploration of this new domain implies basic research while keeping a potential product in mind. Thus, beyond a scientific proof of concept, our navigation methods have to achieve a good cleaning performance under various real-world conditions occurring in typical apartments.

The particular challenges of cleaning-robot navigation result from the following constraints, which domestic floor-cleaning robots are subject to [157, 518, 520]: (i) they should systematically cover their workspace while avoiding both repeated coverage and uncleaned areas, (ii) they should autonomously accomplish their task without any user intervention, (iii) their application should not require modifications of the environment, or —in case modifications are needed— only such modifications, which would also be required for traditional vacuum cleaners, (iv) they should be usable "out of the box" by a wide target group including technically unexperienced users, (v) they should be feasible for everyday usage of a consumer, and (vi) their price should not exceed that of a traditional vacuum cleaner. As a first step towards a full-fledged cleaning robot for household usage, this thesis proposes essential navigation strategies.

1.2. Background of Proposed Work

The particular challenges of cleaning-robot navigation introduced in section 1.1 can only be solved by choosing appropriate building blocks, upon which the proposed navigation strategies are built. On the hardware level, these building blocks are our *custom-built cleaning robot* and *omnidirectional vision* as primary sensory input; on the level of navigation strategies, the proposed algorithms rely on *meandering lanes* as motion strategy, *dense topo-metric maps* as spatial representation, and on *local visual homing* for estimating spatial relations.

Custom-Built Cleaning Robot

Commercially available cleaning robots (section 2.2.1) are differential-drive robots (textbook: [586]) and usually have a circular shape with a diameter of approximately 30 cm and a height of approximately 10 cm. Domestic floor-cleaning robots are consumer goods, and their price should not exceed that of a traditional vacuum cleaner. This limits the costs for the robot's on-board computer and sensory equipment [I71]. Cleaning robots can therefore only be equipped (i) with a small number of cheap sensors, (ii) with limited computational power, (iii) and with restricted memory capacity. These limitations have to be considered for developing navigation strategies suitable for a domestic floor-cleaning robot.

For testing our navigation strategies, a mobile-robot platform with dimensions and properties similar to other cleaning robots is best suited. Using a commercially available platform turned out to be not appropriate for our needs. On the one side, existing commercial *cleaning robots* do not allow low-level control of the robot as is required for our experiments. On the other side, for available *research platforms* low-level control is possible, but they either differ in the robot's geometry and size or they do not allow to integrate an omnidirectional vision setup into the robot's housing. We therefore built[1] a research prototype which fits our requirements. Some time after this decision, the Yujin iClebo kobuki [I111] and the iRobot Create [I54], two research platforms with the shape and size of floor-cleaning robots, became commercially available.

At the current point in time, we do not execute our navigation strategies on-board, but rather use a client-server framework for off-board processing. The robot's sensory data is transferred to an external host computer which executes the navigation algorithms and sends back a motion command to the robot. This approach is required because we decided to implement the proposed navigation strategies with our existing rapid-prototyping framework, which cannot be executed on the robot's computer because it requires more computational power than is available on-board. We favor our rapid-prototyping framework because it allows to rapidly implement and test a large number of different approaches or parameter combinations. By this means, we avoid the implementation effort required for an optimized implementation of our methods executable on the robot's on-board computer. Once the most promising approaches are identified, we can then optimize our methods for the target hardware of a potential product.

Omnidirectional Vision as Primary Sensory Input

As main sensor, we decided to rely on an omnidirectional vision setup which acquires panoramic images with a horizontal field of view of full 360°. We consider omnidirectional vision setups to be an appropriate sensor for a domestic floor-cleaning robot because they (i) are relatively cheap sensors, (ii) provide dense sensory information with a large field of view, and (iii) are multi-purpose sensors which can —beyond navigation— also be used for various other purposes outside the scope of this thesis.

Our custom-built cleaning robot is currently equipped with an omnidirectional vision setup beyond the budget of a potential product. By relying on a better sensor than the one we would use for a product, we reduce the sensor's influence onto the performance of the proposed navigation methods. In case the proposed navigation strategies do not yield good results, it is likely that these results are due to a weakness of the proposed method rather than to the available sensor data. Once

[1]The mechanical construction was mainly pursued by Klaus Kulitza, and Martin Krzykawski was responsible for the software integration. Lorenz Hillen contributed conceptually to questions regarding the omnidirectional vision setup and —to some minor aspect— regarding other sensors.

the navigation methods operate accurately and reliably, we will work towards using a sensor which can also be used in a potential product (section 7.3).

Meandering Lanes as Motion Strategy

While early cleaning robots relied on random walk or preprogrammed motion patterns, nearly all recent robots cover their workspace systematically by multiple segments of meandering lanes[2] (table 2.2). Meandering lanes are straight lanes placed besides each other at a predefined and constant distance. They allow for efficiently covering the area accessible to the robot while avoiding both repeated coverage and uncleaned areas. For this reason, our navigation strategies also rely on meandering lanes and a decomposition of the robot's workspace into segments of meandering lanes.

Dense Topo-Metric Maps

A further aspect of efficient coverage includes to distinguish already cleaned areas from uncleaned areas. This is usually accomplished by mapping the robot's environment (e.g. [518]). The map of the robot's environment is built during the cleaning process and forms the core of any systematic navigation strategy. Besides for distinguishing cleaned from uncleaned areas, it is used for guiding the robot along parallel cleaning lanes, for path planning, and for path following. We consider systems relying on a predefined map, which needs to be installed on the robot, to be not suitable for a consumer robot.

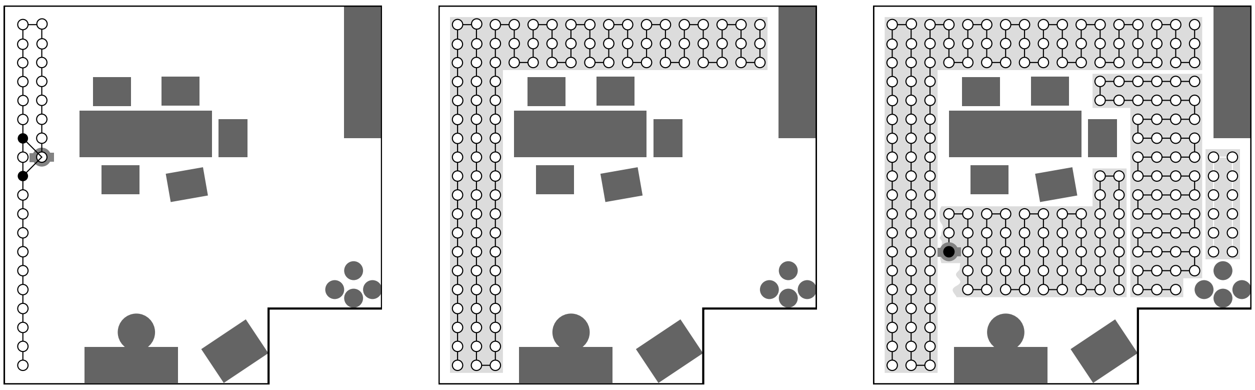
The crucial questions related to mapping are (i) how to relate the perceived visual information with a position in space and (ii) how to integrate this information into a map representing several places. A parsimonious way of characterizing a place is to store the raw sensory information, i.e. the panoramic image itself without detecting visible features perceived when visiting the place (reviews: [189, 403, 642]). By this means, images acquired at former robot positions are used as landmarks[3] and to represent already cleaned places. A straightforward approach to mapping is then to represent several places by graph nodes and to link adjacent places if they are directly reachable from each other. Our particular task requires (i) some sort of distance information to keep the robot at a constant distance from its previous lane, and (ii) a fine spatial resolution for precise navigation. Aspect (i) can be resolved by attaching position information to the place nodes; aspect (ii) can be solved by a grid-like distribution of place nodes, which represent nearby positions in space. By this means, dense topo-metric maps are obtained which can be easily built from the available visual information and allow for efficient map operations even with limited hardware. The application of dense topo-metric maps is currently restricted to simple map-building and localization tasks [171, 271, 377, 426, 505, 506, 538]; our navigation strategies are the first attempt to apply such maps for a more sophisticated task.

Local Visual Homing

Local visual homing methods are a main area of expertise of our research group. From two images acquired at nearby positions in space, homing methods estimate the rotation and the direction, but not the absolute length, of the robot's motion between the positions of image acquisition. Hence they can be considered as partial solutions to the general problem of ego-motion estimation (textbook: [641], review: [130, 191, 563]). In its original sense, local visual homing is the capability of a robot to return to a previously visited place under visual control. In the context of this thesis, local visual homing is applied for estimating angular relations between two places stored in the map without

[2]The term "meandering lanes" is used in this context to describe the robot's motion pattern along alternating lanes; it should *not* suggest that the robot's motion is aimless or undirected.

[3]The term "landmark" refers to a cue used for navigation. Such cues can in the context of this thesis be entire omnidirectional images acquired at former robot positions, objects visible in the omnidirectional images, or features detected by point of interest operators (and potentially associated with an estimate of their position in space).



(1) Trajectory controller for guiding the robot along meandering lanes

(2) Dense topo-metric map representing the cleaned area (edges omitted)

(3) Loop-closure detection to determine if the robot approaches an already cleaned area

Figure 1.1.: Navigation strategies considered in this thesis. The subfigures (1) to (3) depict the core ideas of these strategies.

physically approaching them. By combining estimates of the angular relations obtained from visual homing and odometry-based distance estimates, metrical position estimates of the current or of former robot positions can be obtained. We consider homing methods to be an appropriate building block for cleaning-robot control because they are parsimonious yet accurate algorithms for partial ego-motion estimation (section 3.5). In the context of this dissertation, we solely apply homing methods without proposing new or improving existing methods.

1.3. Objectives of This Thesis

As a complete control scheme required for a full-fledged cleaning robot is beyond the scope of this thesis, we restrict ourselves to two essential substrategies of such a control scheme: (i) *vision-based trajectory control and mapping* (section 1.3.1) and (ii) *visual detection of already cleaned areas* (section 1.3.2). These substrategies (figure 1.1) rely on the building blocks introduced in section 1.2 and are —to the best of our knowledge— the first application of these building blocks for the control of floor-cleaning robots. Thus, this thesis is also a feasibility study for the applicability of the building blocks for our particular application.

1.3.1. Vision-Based Trajectory Controller and Mapping

We propose a mostly vision-based trajectory controller guiding the robot along parallel and meandering lanes while concurrently building a dense topo-metric map of its environment (figures 1.1.1 and 1.1.2). To extend its dense topo-metric map, the robot successively adds snapshots taken at regular distances. On the subsequent lane, these snapshots are used to estimate the robot's current distance to the previous lane. This involves taking the bearing from the robot's current position to at least two snapshots (figure 1.1.1; filled circles) stored along the previous lane by applying local visual homing. By fusing bearing and compass estimates with an odometry-based estimate of the distance between the two considered snapshots, the robot's current distance to the previous lane can be computed by triangulation. In contrast to traditional mapping methods, we do not compute the robot's full pose w.r.t. an external frame of reference. We rather rely on partial pose estimation and only compute the necessary and sufficient information required to solve the task. For our particular method, this includes estimates of (i) the robot's distance to the previous lane and (ii) the robot's orientation w.r.t. world coordinates. These estimates are used to keep the robot at a constant and predefined distance to its previous lane. The rectangular areas covered by the proposed method can be combined by more advanced cleaning strategies (not considered in this dissertation) to completely cover complex-shaped workspaces.

1.3.2. Visual Detection of Already Cleaned Areas

Reliably detecting already cleaned areas is an important prerequisite for an autonomous cleaning robot in order to avoid both uncleaned areas and repeated coverage (figure 1.1.3). This problem is also referred to as loop-closure problem and usually occurs in the context of map-building when the robot approaches an already mapped area (textbooks: [110, 586, 630], reviews: [25, 74, 150, 175, 432, 631]). Loop-closure detection requires to visually compare the robot's current camera image to images stored in the dense topo-metric map. The detection of loop closures has to be purely vision-based, because the robot's position estimate could drift from the robot's true position. Due to their fine spatial resolution, loop-closure detection in dense topo-metric maps is particularly challenging: (i) the large number of images stored in the map requires efficient image comparison techniques and (ii) only retrieving exactly the image from the map which is spatially closest to the robot's current position can avoid repeated coverage or gaps between lanes. We propose two approaches for loop-closure detection, which we refer to as *holistic* and *signature-based* approach.

1.3.2.1. Holistic Loop-Closure Detection

Such methods detect loop closures by a pixel-by-pixel comparison of the entire image followed by a binary classification whether the images were acquired at the same or at different positions in space. Holistic loop-closure detection methods require the images to be aligned w.r.t. a common reference direction. For this purpose, the visual compass method suggested by Zeil, Hoffmann, and Chahl [718] can be integrated into the loop-closure detection process. It step-by-step rotates one of the compared images while keeping the orientation of the other constant, repeatedly compares the shifted image with the other image, and searches for the best match. The residual of the best match is used for deciding whether or not the compared images were acquired at identical positions in space. We propose an accelerated variant of this widely used compass method operating in the Fourier domain. Our method computes the compass estimate without repeatedly shifting and comparing images. As common image comparison functions are not invariant against changes of the illumination, we investigate image preprocessing techniques as a means to increase the robustness against such image disturbances. The goal of this chapter is twofold. First, we seek for combinations of image preprocessing and comparison functions which can efficiently and accurately solve the loop-closure problem for various indoor environments and under different illumination conditions. Second, we systematically assess the compass accuracy of the standard and our accelerated compass method.

1.3.2.2. Signature-Based Loop-Closure Detection

Signature-based approaches to loop-closure detection derive a low-dimensional signature from the entire camera image and compare the resulting signature to the signatures stored in the dense topo-metric map. Hence, they compare signatures instead of comparing images pixel-by-pixel as is the case for holistic methods. The dissimilarity value is used to decide whether or not the compared places are identical. The efficiency of signature-based approaches results from the low dimensionality of the signatures and from their rotational invariance. Thus, signature-based methods do not require the application of a visual compass prior to image comparison as is the case for the holistic methods. Tolerance against changes of the illumination can for signature-based approaches be achieved by certain combinations of signature and comparison functions. For signature-based methods, we do not consider image preprocessing techniques because we expect the tested methods to be sufficiently tolerant against such changes. The objective of the chapter is to find a combination of a signature and a comparison function which can be computed efficiently and which accurately detects loop-closures for different environments and under different illumination conditions.

1.4. Outline of This Thesis

This thesis is divided into two parts. The first part (chapters 2 and 3) introduces relevant concepts and reviews related work. The second part (chapters 4 to 6) describes the proposed navigation strategies for trajectory control and mapping (chapter 4), for holistic loop-closure detection (chapter 5), and for signature-based loop-closure detection (chapter 6).

Chapter 2: Cleaning Robots
In this chapter, we put autonomous floor-cleaning robots for domestic usage into the context of other cleaning robots and —more general— of mobile service robots. We give an overview over commercially available floor-cleaning robots and over related academic research. We then discuss the relevance of the reviewed aspects for our work.

Chapter 3: Visual Navigation Based on Omnidirectional Images
We briefly introduce omnidirectional vision and present a detailed literature review on navigation methods for wheeled mobile robots relying on omnidirectional images as primary sensory information. We analyzed the available literature and propose a categorization of navigation methods operating on omnidirectional images which is used to relate our contributions to existing work in this field.

Chapter 4: Trajectory Controller Based on Partial Pose Estimation and Dense Topo-Metric Maps
We propose a navigation strategy for covering a rectangular area with meandering cleaning lanes while concurrently building a dense topo-metric map of the robot's environment. We describe how dense topo-metric maps are applied by this method and give a mathematical derivation of the proposed control algorithm relying on omnidirectional images and partial pose estimation. To assess the performance of our controller, we furthermore propose performance measures which are applied to data obtained from real-robot experiments.

Chapter 5: Holistic Loop-Closure Detection and Visual Compass
In this chapter, we suggest holistic methods for loop-closure detection, which include image preprocessing to increase robustness against illumination changes, a visual compass to align images, and image dissimilarity functions to measure the similarity of the considered images. We describe the tested preprocessing methods and image dissimilarity functions, and we derive the Fourier-based compass variant. Database experiments are conducted (i) to identify the most suitable combination of preprocessing method and image dissimilarity function for loop-closure detection and (ii) to assess the accuracy of the visual compass.

Chapter 6: Signature-Based Loop-Closure Detection
The objective of this chapter is to evaluate signature-based approaches to loop-closure detection in different environments and under changing illumination conditions. To systematically test a large number of signatures and signature-comparison functions, we conducted database experiments. For the combination of signature and signature-comparison function yielding the best results in the database experiments, real-robot experiments were conducted.

Chapter 7: Overall Summary, Discussion, and Outlook
We summarize and discuss the results of chapters 4 to 6 and put them in a broader context. Beyond that, future working directions resulting from this thesis are proposed.

Appendix
Additional data and mathematical derivations are given in the appendices (appendices A to D) of this thesis.

2. Cleaning Robots

We introduce cleaning robots as a subdomain of mobile sevice robots. Furthermore, we give a market survey of commercially available cleaning robots and a literature survey on related research.
The remainder of this section is structured as follows: section 2.1 introduces the field of cleaning robots as a subdomain of mobile service robots. The particular properties of domestic floor-cleaning robots are then discussed in section 2.2 including reviews on commercially available robots (section 2.2.1) and on academic research in this field (section 2.2.2). The section ends with a short discussion of the relevance for our work (section 2.3)
This chapter extends the introductions of our recent journal publications (sections 3 of Gerstmayr-Hillen *et al. [222] and section 1.1 of* Möller *et al. [457]).*

2.1. Cleaning Robots as Subdomain of Mobile Service Robots

Cleaning robots are considered a sub-domain of mobile service robots (figure 2.1) assisting humans in monotonous and tedious tasks [209, 562, 580] and are both subject of academic research and commercially available. Thus, the surveys on related work presented in this section always contain examples from both fields. The large interest in cleaning robots is probably due (i) to the large market potential of consumer and professional products and (ii) to the variety of possible application

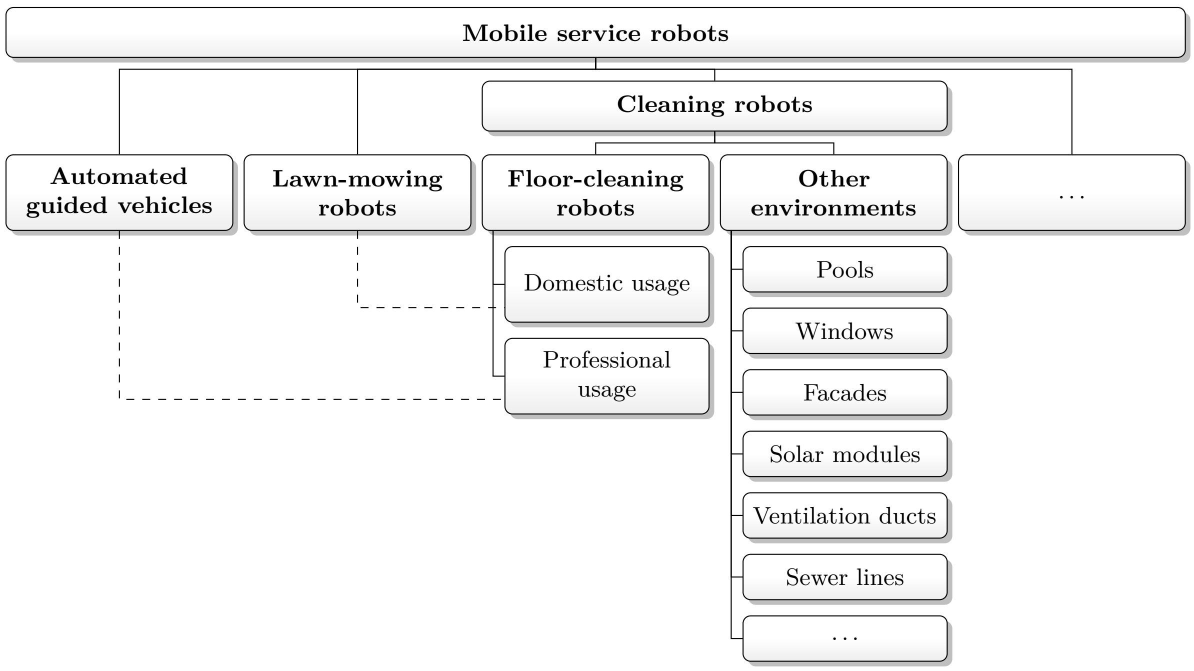


Figure 2.1.: Placement of domestic cleaning robots in the domain of mobile service robots. Closely related application domains are linked by dashed lines. Extended after [157, 209, 562, 580].

Table 2.1.: Examples of mobile service robots related to domestic floor-cleaning robots. The subtables list examples of cleaning robots for other environments than floors (subtable (1)), professional floor-cleaning robots (subtable (2)), and mobile service robots closely related to floor-cleaning robots (subtable (3)). The lists in the subtables are not exhaustive.

(1) Cleaning robots for other environments (i.e. except for floors)

Application domain	Academic	Commercial
Pools	[589]	iRobot: Verro series [I55] Zodiac: Polaris 9300 and 9300xi [I114, I115]
Windows & facades	[88, 207, 445, 525, 556, 579] Fraunhofer: SIRIUSc [155] Various[1]: Sky Cleaner series [721]	ALM Robotics: Windoro [I10] Ecovacs: Winbot W353 and W553 [I20] Serbot Innovations: Gekko Plus [I86]
Solar modules	—	Serbot Innovations: Gekko series [I84, I85, I87] Solarbrush: L and H [I88, I89]
Ventilation ducts	[174, 281, 335, 696, 708]	Cyclone Ventilation: CYBOT [I17] Danduct: Icetech and Multi Purpose Robot [I18, I19]
Sewer lines	[283, 539, 591] Fraunhofer: Spy system [154, 156]	IBG Hydro-Tech: Hydrocut series [I47] Ka-Te: grinding robot series [I94]

[1] Cooperation project of the University of Hamburg (Germany), BeiHang University (Beijing, China), and the City University of Hong Kong.

(2) Professional floor-cleaning robots

Academic	Commercial
[111, 203, 208, 290] East Japan Railway Company: Various robots [707]	Floorbotics: IVAC [I25] Fuji Heavy Industries: Subaru RFS1 [I29] Intellibot: HydroBot, AeroBot, and DuoBot [I51–I53] Robosoft: AutoVac 6 [I80]

(3) Other mobile service robots related to floor-cleaning robots

Application domain	Academic	Commercial
Autonomous/Automatic Guided Vehicles (AGVs)	[112, 330, 414, 636, 646, 689, 722] Review: [672]	FROG AGV Systems: Various robots [I28] Hi-Tech Group: Intellicart series [I95] JBT Corporation: Various robots [I58] Jervis B. Webb Company: Various robots [I59] egemin Automation: Various robots [I21]
Lawn mowing	[284, 526, 643, 684]	Gardena: R40Li [I30] Friendly Robotics: Robomow series [I26] Husqvarna: Automower series [I46] Zucchetti: Ambrogio series [I116] Bosch: Indego 10 [I1, I16]

areas. These range from apartments to large buildings or factory spaces to hazardous environments or environments difficult to access [157, 176, 209, 518, 519, 562, 580].

Academic research on cleaning robots frequently uses experimental robot platforms and focuses on navigation strategies (including localization, mapping, and planning of cleaning paths), kinematics, control, and sensor data integration. Beyond that, companies have to incorporate aspects of usability, safety issues, production engineering, marketing, and economic feasibility for designing their cleaning robots. The navigation strategies presented in chapters 4 to 6 arose from searching a real-world application as testbed for local-visual homing methods. With our research on cleaning-robot navigation, we try to bridge the gap between basic research and application-oriented research. Therefore, aspects such as customer demands, characteristics of a consumer product, and aspects of industrial product development strongly influence our research.

The navigation strategies of all cleaning robots have in common that they aim at complete coverage of the accessible workspace without leaving areas uncleaned and, ideally, without visiting areas multiple times. Especially if the robot is powered by batteries, it is essential to keep the proportion of repeated coverage as small as possible. Depending on the surface to be cleaned, floor-cleaning robots and robots for cleaning other surfaces can be distinguished [157, 518]. Cleaning robots for other surfaces have to navigate e.g. on strongly inclined surfaces or under water. Therefore, they require drives and safety precautions specialized for the specific task. These robots are usually remotely operated, with power and cleaning medium being provided over cables and hoses, respectively (ELKMANN, HORTIG, and FRITZSCHE [157]). Example workspaces of such robots include pools, windows, facades, solar modules, ventilation ducts, and sewer lines (table 2.1.1).

Floor-cleaning robots are usually wheeled robots. In order to operate autonomously, the robots are equipped with batteries, computers, dirt reservoirs, brushes, suction units, or tanks for detergent. Such robots can be further categorized into robots for domestic and professional usage (ELKMANN, HORTIG, and FRITZSCHE [157]). Professional systems (table 2.1.2) are used for cleaning of large buildings such as stations, airports, shopping centers, or factory spaces [157, 176]. Robots for professional usage are large and heavy-weight machines which can carry a considerable payload (e.g. tanks for water, detergent, or waste water). They are usually equipped with strong batteries, large computational power, and sophisticated sensor systems. For navigation, such systems can rely on artificial beacons or on strategies which require the footprint of the environment to be known beforehand. With these properties, professional floor-cleaning robots rather resemble autonomous guided vehicles (AGV, e.g. [41], table 2.1.3) than domestic cleaning robots. Since the focus of this thesis is on domestic floor-cleaning robots, these robots will be described in more detail in the following section.

2.2. Domestic Floor-Cleaning Robots

In order to clean a complex-shaped workspace such as a room or an entire apartment, domestic cleaning robots need to be small and agile [176, 518, 519]. As their price should be comparable to the price of a standard vacuum cleaner, domestic cleaning robots can only be equipped with little battery power, low computational power, and a small number of cheap sensors [I71]. The robots should be applicable "out of the box" without knowing the footprint of the workspace before cleaning, with as little modifications to the environment as possible, and in the ideal case without user intervention. The third aspect is essential if the robot is supposed to clean if the user is away from home. It includes that the robot should not get stuck underneath obstacles like furniture. Furthermore, the robot should autonomously return to its docking station for recharging or after cleaning. Domestic floor-cleaning robots have to be considered consumer goods. For this reason, they are closer related to lawn-mowing robots (table 2.1.3) than they are to professional floor-cleaning robots or to cleaning robots for other environments. Such robots are both available on the market and the subject of academic research; these two fields will be reviewed in the remainder of this section. Since details about commercial cleaning robots are rarely published in scientific literature, links to the manufacturers' web pages are given instead; patents were not been considered in this thesis.

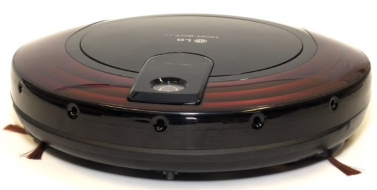
2.2.1. Commercial Products

Commercial floor-cleaning robots for household usage have been available on the market since approximately the year 2000. Table 2.2 lists some of these robots, and three currently available

Table 2.2.: Examples of commercially available floor-cleaning robots. The given list is not exhaustive; extended after [I11, 578].

			Modes				Sensors									
	Manufacturer	Model	Spot	Random	Wall	Maeander	Bumper	Cliff	Distance	Laser range	Camera	Floor camera	Gyroscope	Accelerometer	NorthStar	Reference
First generation	Agait	E-Clean series	✓	✓	✓	✓	✓	✓								[I2–I5]
	Agama	AiBot 300 series		✓	✓		✓									[I6, I7]
	Agama	AiBot 500 series	✓	✓	✓	✓	✓	✓								[I8, I9]
	Evovacs	Deebot series	✓	✓	✓		✓	✓	✓							[I99–I103]
	Electrolux	Trilobite 2.0		✓	✓				✓							[518]
	iRobot	Roomba series	✓	✓			✓	✓	✓							[I56, 639]
	iRobot	Scooba series		✓			✓	✓	✓							[I57]
	Kärcher	RC series		✓			✓	✓	✓							[I60, I61]
	Mamirobot	Sevian series	✓	✓	✓		✓	✓	✓							[I65]
	Yujin	iClebo home	✓	✓			✓	✓	✓							[I110]
Second generation	Evolution Robotics	Mint series		✓		✓	✓	✓	✓						✓	[I23, I24, I71]
	Hanool	Ottoro series	✓	✓		✓	✓	✓	✓				✓			[I104, I105]
	LG	Hom-Bot series	✓			✓	✓	✓	✓		✓	✓	✓	✓		[I63, I64]
	Neato Robotics	XV series				✓	✓	✓		✓				✓		[I72–I75]
	Moneual	Rydis R750	✓			✓	✓	✓	✓				✓	✓		[I70]
	Philips	HomeRun FC9910	✓			✓	✓	✓	✓		✓		✓	✓		[I77]
	Samsung	Navibot Series	✓			✓	✓	✓	✓		✓		✓	✓		[I81, I82]
	Vorwerk	Kobold VR100				✓	✓	✓		✓				✓		[I106]
	Yujin	iClebo smart/arte		✓		✓	✓	✓	✓		✓		✓	?		[I109, I112]

(1) Samsung NaviBot Silencio SR-8895


(2) LG Hom-Bot 2.0

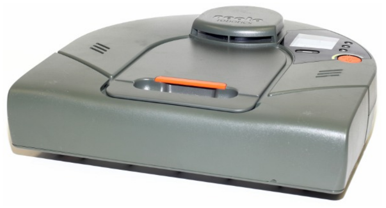
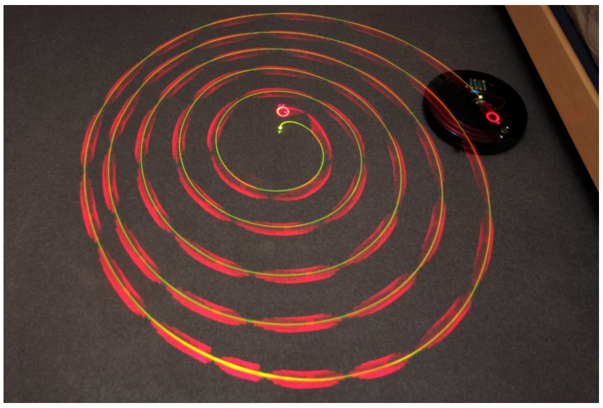

(3) Neato XV-11

Figure 2.2.: Examples of commercially available floor-cleaning robots for household usage. Photos by Lorenz Hillen. Figure best viewed in color.

cleaning robots are shown in figure 2.2. All the robots we are currently aware of are differential-drive robots (textbook: [586]). Most of the robots have a circular shape with a height of approximately 10 cm and a diameter of approximately 30 cm. Exceptions regarding the robot's shape include (i) the Neato XV series ([I72–I75] and figure 2.2.3) and the Vorwerk Kobold VR100 [I106] which are D-shaped and (ii) the LG Hom-Bot Square [I64] which is squared with rounded corners. Depending on their navigation strategy, they can be categorized into *first-generation* and *second-generation* products.

2.2.1.1. First-Generation robots

First-generation robots rely on preprogrammed movement patterns such as spirals (figure 2.3.1) or on random-walk strategies (figure 2.3.2). Thus, achieving complete coverage requires a relatively


(1) Spiraling movement pattern. The trajectory shown in this image was obtained by the LG Hombot 2.0 operating for 77 s in its spot cleaning mode. To trace the robot's trajectory, a green light-emitting diode (LED) was mounted on the robot resulting in the thin green uninterrupted trace. The red trace was caused by the robot's blinking control elements.

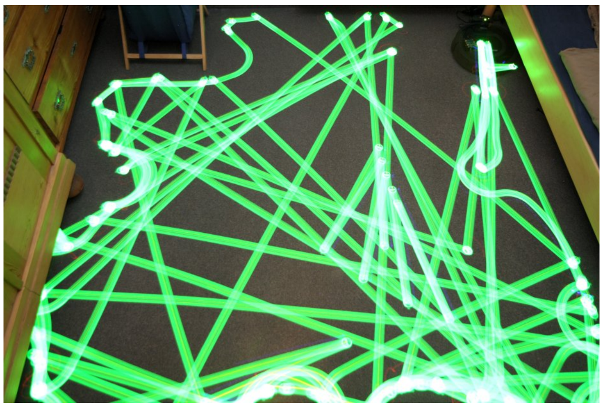
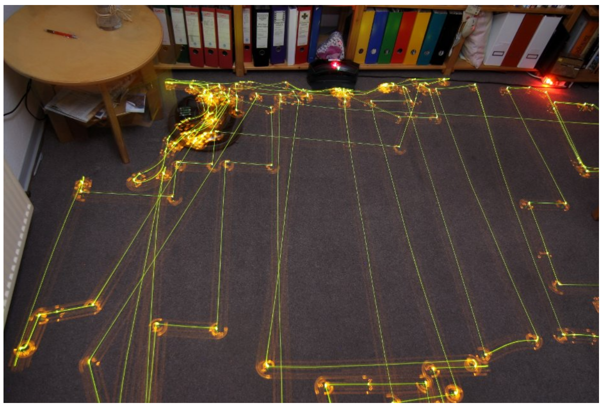

(2) Random walk obtained by the iRobot Roomba 770. The robot was released from its docking station (top right) and returned to it after 689 s. The thick green trace results from the robot's control elements.

Figure 2.3.: Navigation strategies of first-generation cleaning robots. Subfigures (1) and (2) show spiraling movement patterns and random walk strategies, respectively. The shown pictures are long-time exposures taken while the robot was operating in the dark; the environment was made visible by flashing with second-curtain synchronization. Thus, the photos depict the situation at the end of the cleaning run. Perspective and lens distortions were not corrected. Photos by Lorenz Hillen. Figure requires color printing.


(1) Samsung NaviBot SR-8895 Silencio. Exposure time: 658 s. The robot was released from the charging station (top center with small red light), but it did not return.

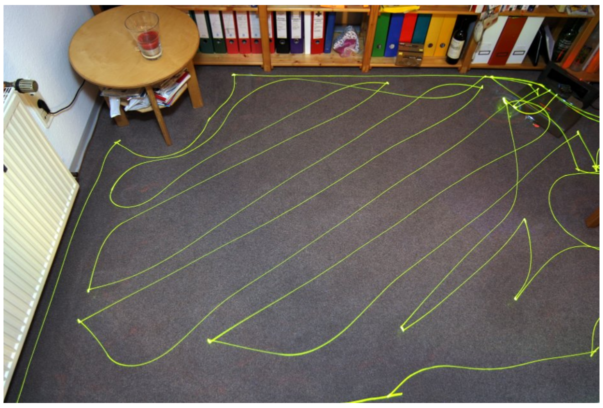

(2) Neato XV-11. The robot was released from its charging station (top right) and successfully returned to it after 448 s.

Figure 2.4.: Navigation strategies of second-generation cleaning robots. The subfigures visualize similar movement strategies by different commercially available cleaning robots. The shown pictures are long-time exposures taken while the robot was operating in the dark; the environment was made visible by flashing with second-curtain synchronization. Thus, the photos depict the situation at the end of the cleaning run. To trace the robot's trajectory, a green light-emitting diode (LED) was mounted on the robot resulting in the thin green uninterrupted trace. Perspective and lens distortions were not corrected. Photos by Lorenz Hillen. Figure requires color printing.

large amount of time and results in a large proportion of repeated coverage [501, 535]. With their limited sensor equipment and with limited computational power, these robots are not capable of building a representation of their workspace. Thus, they cannot distinguish covered areas from areas which still need to be cleaned (Prassler and Kosuge [518]). Several robots of this class were or are still available on the market (table 2.2, upper part). Among all these robots, the family of iRobot's Roomba robots is the most well known and probably the most frequently sold domestic cleaning robot.

2.2.1.2. Second-Generation robots

Second-generation robots (table 2.2, lower part) rely on systematic exploration strategies and can recognize and directly approach uncleaned areas. By this means, they can cover the workspace more efficiently and with a smaller proportion of repeated coverage. Second-generation robots move along meandering and parallel lanes (figure 2.4), and complete coverage is achieved by combining several segments of meandering lanes. To accomplish their task, the robots are —in addition to the basic sensors also used by first-generation robots— equipped with more elaborated sensors such as cameras, gyroscopes, accelerometers or laser-range finders. Although little is known about the navigation strategies of these robots, it is likely that they build a map of their workspace for achieving the cleaning task. Second-generation robots have been available since 2008 and are equipped with a 360° laser range finder (e.g. Neato XV series [I72–I75], Vorwerk Kobold VR100 [I106]), with a combination of a gyroscope and a monocular camera directed towards the ceiling (e.g. LG Hom-Bot series [I63, I64], Yujin iClebo smart and arte [I109, I112], Samsung Navibot SR-8855 and SR-8895 Silencio [I81, I82], Philips HomeRun FC9910 [I77]), or with a gyroscopes as primary sensor (e.g. Moneual Rydis R750 [I70]). The navigation strategies described in chapters 4 to 6 are essential building blocks for systematically covering complex-shaped workspaces by combining several cleaning segments of parallel lanes. To this end, our navigation strategies are closely related to second generation robots.

2.2.2. Academic Research

Academic research on autonomous cleaning robot dates back to the 1990's (seminal workshop proceedings: [49], reviews: [176, 519]). In 2002, the first "International Contest for Cleaning Robots" was held jointly with the "IEEE International Conference on Intelligent Robots and Systems" (IROS, [I92, I107]). The test arena was a furnished single-room apartment with a base area of 25 m^2. The goal of the competition was to completely clean the accessible area within 10 min while avoiding both obstacle collisions and interventions of the operator. In the contest, 12 teams from different universities competed. Most of the robots were custom-built with different sensory equipment and relied on random or systematic cleaning strategies (Prassler, Hägele, and Siegwart [521]). The results of the competition revealed that the cleaning performance was only moderate. Many robots required human intervention because they got stuck in corners or underneath obstacles. The competition remained a unique event. Thus, the chance was missed to establish a possibility for benchmarking and a platform for knowledge exchange like the RoboCup [I96] is in the fields of robot soccer and rescue robotics.

Since the cleaning-robot contest, autonomous cleaning robots have only received little attention in academic research. This is in contrast to (i) the fast advances of the commercial sector and (ii) to the tremendous achievements in the field of autonomous navigation (chapter 3). Nevertheless, we are currently not aware of an academic paper describing an entire robot system capable of autonomously cleaning a complex-shaped workspace. Several aspects were investigated in academic research including *planning of cleaning paths* and *simultaneous localization and mapping* (SLAM). As these aspects are related to the work presented in this thesis, they will be briefly discussed in the following; further aspects which are not relevant for this thesis include human robot interaction (e.g. [185]) or hardware design of cleaning robots (e.g. [499]).

2.2.2.1. Cleaning Path Planning

In the domain of cleaning path planning, *complete coverage planning algorithms* can be applied for the navigation of an autonomous cleaning robot. These methods plan paths in order to completely cover the robot's workspace while minimizing the portion of repeated coverage (Choset [109]). Complete coverage algorithms can be categorized into methods assuming the map of the environment to be known a-priori (e.g. [107]) and into methods including map building (e.g. [362, 382, 391]). For details on these methods, the reader is referred to the reviews [109, 362, 501]. In contrast to complete coverage planning, *frontier-based planning methods* release the restriction of physically visiting every accessible place. These methods solely aim at complete *sensor* coverage for mapping the robot's workspace by planning paths in order to approach free space at the border of explored areas (e.g. [193, 710, 711]).

2.2.2.2. Simultaneous Localization and Mapping for Cleaning Robots

SLAM algorithms concurrently localize the robot within a map while building a map of the robot's environment. By integrating sensor data over time, the map is iteratively updated —usually by applying a Bayesian filter framework. For details on such methods see the textbooks [110, 586, 630] and section 3.6 of this thesis. In the context of cleaning robots, SLAM methods relying on *vision* or *other sensor modalities* as primary source of information are applied.

Visual SLAM

The following visual SLAM methods all rely on a monocular camera directed towards the ceiling and cover the robot's workspace with meandering lanes. With respect to the taxonomy of mapping methods proposed in section 3.6, these methods build sparse model-based maps (section 3.6.3.1). The methods proposed by Jeong and Lee [309, 310] use an extended Kalman filter (EKF) for estimating the robot's pose and the position of the landmarks. As landmarks, they rely on SIFT features (Lowe [389]) and on SIFT features combined with line features (see section 3.3.1.3 for a brief description of feature detectors and descriptors). The latter method was extended to multi-robot SLAM using a particle filter for state estimation (Lee and Lee [360, 361]). The focus of these papers is on map building and on obtaining an estimate of the robot's pose; further aspects of cleaning strategies for covering workspaces with a complex layout are not described. Nevertheless, the similarity between figures in the publications [360, 361] and a video [I108] available in the internet suggest that the Samsung robots ([I81, I82] and table 2.2) use SLAM algorithms developed by this research group. The SLAM method proposed by [106] relies on a monocular camera directed towards the robot's movement direction. It estimates the robot's current state and the 3D-positions of the known feature points (detected by the Harris corner detector; Harris and Stephens [277]) by an extended Kalman-filter framework. New features are added without delayed measurement by assuming a large initial uncertainty along the feature's viewing direction. Later on, the initial uncertainty is refined. The method by Kwon, Song, and Kang [350] is an algorithm for Monte-Carlo localization (section 3.3.2.1) tested in the context of cleaning robot navigation. It relies on a known database of low-dimensional feature descriptors for corner features with known positions sensed on the ceiling of the robot's workspace. In principle, the method could be extended to a feature-based SLAM method like the methods described above.

Other Sensor Modalities

The method by Gutmann et al. [263, 264] is referred to as vector-field SLAM: it learns the spatial variation of a continuous signal and uses the signal variations for position correction by an extended Kalman filter, by an exactly sparse extended information filter (ESEIF), or by graph-based optimization techniques (see section 3.3.2.1 for a brief description of position estimation techniques).

As external reference signal, the robot derives bearing information from static IR spots projected onto the ceiling by the Northstar system (Yamamoto et al. [709]). A similar approach referred to as magnetic field-based SLAM is used by Vallivaara et al. [656, 657] where the robot learns a map of anomalies of the ambient magnetic field arising in indoor environments. The map is approximated by Gaussian process regression (textbooks: [46, 532]), and a particle filter is used to estimate the robot's state. The authors conclude that the obtained maps are stable over time and that the resulting localization accuracy is sufficient for the control of an autonomous floor-cleaning robot.

2.3. Relevance for Our Work

The market survey in section 2.2.1 clearly reveals that none of the currently available cleaning robots is equipped with an omnidirection camera system. Related second-generation robots rely on laser-range finders, gyroscopes and accelerometers, or a monocular camera directed towards the ceiling. Nevertheless, we consider omnidirectional vision to be an appropriate sensor modality for such robots (please refer to section 3.2.4.2 for a detailed discussion on this issue). Because of being the first application of omnidirectional vision for cleaning-robot control, this dissertation has the character of a feasibility study.

Regarding the related work of the academic domain, the approaches reviewed in section 2.2.2 are of limited relevance for our work. Due to being tailored to their particular sensor modalities, *vector-field SLAM* (Gutmann et al. [263, 264]) and *magnetic-field SLAM* (Vallivaara et al. [656, 657]) cannot be applied with our particular robot equipped with an omnidirectional vision setup (section 4.4.3). *Complete coverage* and *frontier-based planning* algorithms both require methods for detecting and approaching free space at the border of explored areas (section 2.2.2.1). Regarding these aspects, they are closely related to high-level navigation strategies of cleaning robots required to completely cover complex-shaped workspaces such as entire rooms or apartments. Such high-level aspects are not considered for this thesis, but they will become relevant for implementing further strategies required to completely cover complex-shaped workspaces (section 7.3.2). Regarding low-level aspects such as mapping and trajectory-control, the methods for complete coverage planning or frontier-based exploration algorithms we are aware of all rely on range information. Range information could in principle be derived by visual stereo computations, but we prefer to solely use visual *intensity* information and avoid the effort for stereo computations. Besides the different sensory information, coverage planning and frontier-based exploration algorithms do not deal with position estimation but assume the position to be known. In contrast, the trajectory controller proposed in chapter 4 does not rely on such an assumption but computes the necessary information from the available sensor information.

Visual SLAM methods relying on a standard pinhole camera facing upwards as described in section 2.2.2.2 can in principle be extended for usage with omnidirectional vision. Nevertheless, we expect them to be computationally more demanding than the methods proposed in this thesis. This is due to (i) using spatial positions of visible image features as landmarks and (ii) to the posterior map corrections inherent to SLAM methods. The former aspect incorporates the computationally demanding steps of detecting features in the image, establishing correspondences, and computing and maintaining estimates of the features' positions in space. With our methods, we try to avoid these steps by using entire images acquired at former robot positions as landmarks (section 3.5.1). Posterior position updates are computationally demanding because their complexity usually grows with the number of positions stored in the map (textbook: [630]). With our methods, we hope to circumvent this step by combining several segments of locally consistent cleaning lanes without enforcing global consistency. Regarding our application, we think that avoiding repeated coverage and uncleaned areas can also be achieved without global consistency but with reliable loop-closure detection (sections 7.2.1 and 7.3.3).

3. Visual Navigation Based on Omnidirectional Images

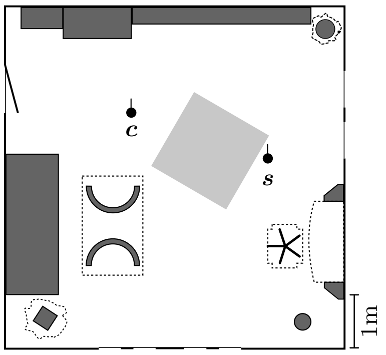
The chapter reviews and categorizes related work on navigation of wheeled mobile robots relying on omnidirectional visual information. This categorization is used to relate the navigation strategies proposed in this thesis to existing works in the research domain.
Section 3.1 briefly motivates this review, and section 3.2 introduces navigation and omnidirectional vision. The proposed categorization of navigation strategies is described in sections 3.3 to 3.6 and includes visual place representation and recognition, visual compass methods, local visual homing, and mapping methods, respectively. The chapter ends with a discussion (section 3.7).
Sections 3.2.4.2, 3.5.1 and 3.6 of this chapter extend sections 3.3.1, 4.1, and 2, respectively, of our recent journal article GERSTMAYR-HILLEN *et al. [222] published in "Robotics and Autonomous Systems". For sake of documentation, sections 3.2.3.2, 3.5.2.3 and 3.7.1.1 briefly mention results which are not considered in this thesis. These include Lorenz Hillen's technical reports [213, 217] and final projects [60, 243, 328, 470, 550, 667, 669] of students supervised by Lorenz Hillen.*
We plan to publish a modified version of the chapter as a review article after the final publication of this dissertation.

3.1. Motivation

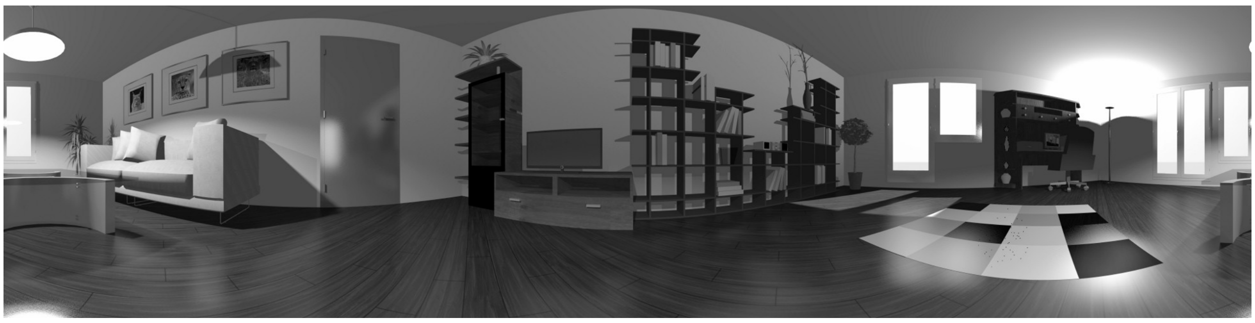
During the last 10 to 15 years, a substantial number of research papers dealing with navigation strategies for wheeled mobile robots relying on omnidirectional vision was published. However, in contrast to the large number of *research papers*, the field is not well covered by *reviews* or *textbooks*: the reviews [52, 99, 130, 139, 175, 189, 191, 432, 563, 642] and the textbook [586] mention omnidirectional vision only as a side issue, while the reviews [25, 74, 150, 194, 341, 631] and the textbooks [41, 110, 630] do not cover omnidirectional vision at all —even though they deal with concepts also relevant for navigation based on omnidirectional vision. We are therefore of the opinion that the field lacks a sound categorization of existing navigation strategies. Thus, we analyzed the available literature and propose such a categorization which is used to relate and compare the navigation strategies proposed in this dissertation. For some groups of navigation strategies, we could reuse or extend existing categorizations, whereas others required to propose a new categorization. Due to the huge amount of original research papers published in the field, we restrict ourselves (i) to papers published since 2000 and (ii) to papers dealing with omnidirectional visual navigation of wheeled robots, i.e. of robots moving in the plane.

Section 3.2 of this chapter defines navigation, introduces prerequisies for navigation and omnidirectional vision, and describes the basic categorization of navigation strategies proposed in this chapter; sections 3.3 to 3.6 then describe the different categories in more detail. To visualize the reviewed strategies, we generated a 3D model of a living room (figure 3.1). Figures 3.10, 3.11, 3.14, 3.20, 3.24 to 3.29, 3.32 to 3.36 and 3.38 show typical properties and situations of the navigation methods as they could be obtained by a robot navigating in the living room. Please note that the

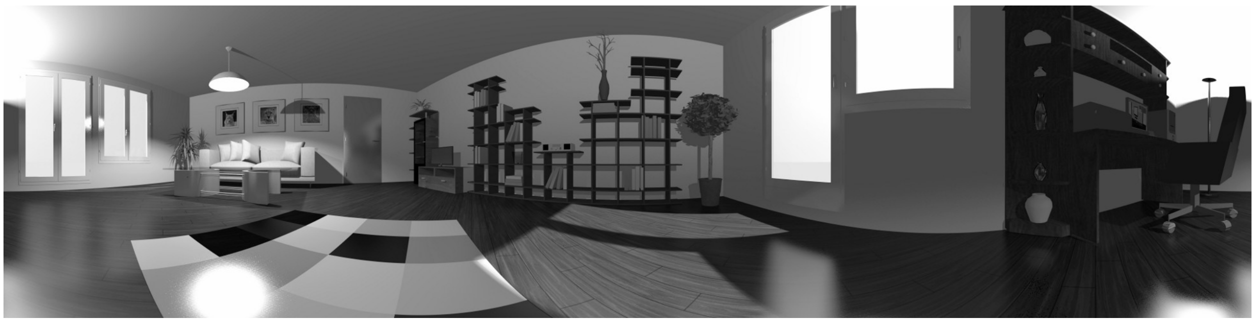
(1) Rendered 3D view



(2) Footprint. Ground-level obstacles are depicted by filled areas. Obstacles above ground level are marked by dashed lines; the carpet is depicted by the light-gray rectangle. The panoramic images were rendered at positions ***c*** and ***s*** with the robot facing the bookshelf (indicated by the little bars).



(3) Rendered example image 1. The robot was placed at position ***c*** facing the bookshelf. For visualizing compass and local visual homing methods (sections 3.4 and 3.5), this image is used as *current view* (i.e. the image acquired at the robot's current position).



(4) Rendered example image 2. The robot was placed at position ***s*** facing the bookshelf. For visualizing compass and local visual homing methods (sections 3.4 and 3.5), this image is used as *snapshot* (i.e. the image acquired at an earlier robot position or the robot's goal position).

Figure 3.1.: Virtual example room for visualization of navigation strategies. The room is used to visualize approaches to characterize places by visual information (figures 3.10, 3.11 and 3.14), visual compass methods (figure 3.20), different groups of local visual homing methods (figures 3.24 to 3.29), and different types of maps (figures 3.32 to 3.36 and 3.38). The 3D model was generated in SweetHome3D [I22] by reusing the furniture models [I32–I40] publicly available from Google 3D Warehouse [I31]. Subfigure (1) gives an impression of the room (rendered using a standard camera), and subfigure (2) shows the room's footprint. The panoramic images depicted in subfigures (3) and (4) were acquired at positions ***c*** and ***s*** (in subfigure (2) marked by filled circles with the bars indicating the robot's orientation).

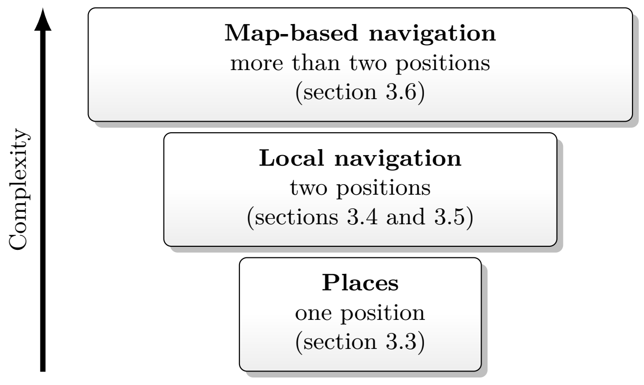


Figure 3.2.: Hierarchy of navigation methods operating on omnidirectional images. The methods' complexity increases from bottom to top, and more complex methods rely on less complex methods. The numbers refer to the sections describing the corresponding level in detail.

figures were manually drawn and not generated by reviewed algorithms operating on rendered image data.

3.2. Introduction

This section introduces basic aspects of navigation and omnidirectional vision necessary to understand the categorization of navigation methods proposed in sections 3.3 to 3.6.

3.2.1. Definition of Navigation

Although being easy to understand, the term navigation is difficult to define and various definitions were proposed. Here, we focus on two definitions: the first is rather technical, whereas the second one is very basic and general. In Levitt and Lawton [367] navigation is defined as the process of answering the following three questions: (i) "Where am I?", (ii) "Where are other places relative to me?", and (iii) "How do I get to other places from here?". This definition is strongly influenced by marine navigation, and it influenced classical attempts to robot navigation. The first question requires the agent —i.e. the animal, human, or robot— to be localized. Thus, its position in the environment has to be known. The second question requires the agent to have knowledge about several places and their positions in space. Such information is typically stored in a map of the environment. The third question is closely related to classical path and motion planning approaches.

In contrast, ethological research revealed that navigation is possible without answering these questions. Therefore, navigation is by Franz and Mallot [189] defined as "the process of determining and maintaining a course or a trajectory to a goal location". According to Franz and Mallot [189], the only requirements for navigating are (i) to move in space and (ii) to recognize the goal. In contrast to the definition of Levitt and Lawton [367], navigation as defined by Franz and Mallot [189] does neither require localization nor a map. Both definitions have in common that navigation behavior needs to be goal-directed. Therefore, the definitions exclude spatial behaviors like obstacle avoidance or course stabilization. Throughout this dissertation, we will follow the more general definition by Franz and Mallot [189].

3.2.2. Basic Categorization of Navigation Methods

For categorizing navigation methods, we propose a three-level hierarchy depicted in figure 3.2. The lowest level of our categorization is the level of *places* and subsumes methods to characterize (i.e. to relate the perceived visual information with a place in space) and to rerecognize places based on visual information. With only a single place, an agent cannot yet navigate, but the navigation methods of the higher levels operate on this place representation to achieve certain navigation capabilities. We will discuss the first level in section 3.3.

The second level is the level of *local navigation*. Local navigation methods operate on visual information obtained at two different but nearby positions in space. Typically, these two places are the robot's current position and a known goal position. Local navigation can be used to guide the robot from its current position towards the goal or to estimate spatial relations between two positions without physically approaching the goal. The navigation strategies of this level are limited to the robot's *sensory horizon*. In case of vision as primary sensory information, this is the area in which the features perceived at the goal position are visible. Local navigation methods are covered in sections 3.4 and 3.5.

The third and most complex level embraces *map-based navigation* strategies, which store several places and their interrelations in a map. For building and operating on the map, map-based navigation methods rely on local-navigation strategies. In contrast to local navigation methods, map-based methods allow the agent to navigate beyond its current sensory horizon. In the review by Franz and Mallot [189], map-based navigation is referred to as way-finding. However, we prefer the term "map-based navigation" because it is more common in the robotics community. Map-based navigation methods are described in section 3.6. This categorization extends the categorizations proposed by Trullier et al. [642] and Franz and Mallot [189], which only include the second and third level of our hierarchy. Before describing the levels, we first introduce essential prerequisites for robust and accurate navigation (section 3.2.3) and introduce omnidirectional vision (section 3.2.4).

The proposed hierarchy focuses on the agent's behavior which results from navigation strategies of each level. Another possible categorization of navigation strategies results from grouping methods depending on whether or not they use metrical position information. Strategies not relying on position information are also referred to as *qualitative* methods, and strategies incorporating metrical position information are referred to as *quantitative* methods. Depending on the used strategy, the agent's pose (i.e. position and orientation) in space or spatial positions of visible image features are computed (section 3.3.2).

Besides the categorization into *qualitative* and *quantitative* navigation methods, *appearance-based methods* embrace navigation strategies which operate solely on image intensity information without estimating spatial positions of visible objects. Thus, appearance-based methods include (i) qualitative strategies and (ii) quantitative strategies if they do not estimate spatial positions of visible features. The term *appearance-based* is borrowed from work on visual object recognition describing approaches making solely use of the object's appearance rather than methods incorporating geometrical object information (seminal paper: Murase and Nayar [471], reviews: [153, 544], textbook: [633]).

3.2.3. Prerequisites for Reliable and Robust Navigation

According to the reviews by [74, 99, 175, 194, 432], a set of four prerequisites can be identified which are crucial for robust and accurate navigation. These prerequisites include (i) *robustness against perceptual aliasing* (section 3.2.3.1), (ii) *robustness against perceptual variability* (section 3.2.3.2), (iii) *correct place recognition* (section 3.2.3.3), and (iv) —in case of map-based navigation— *correct sensor-data integration* (section 3.2.3.4). The first two aspects strongly influence the perceived visual information. In their context, robustness means that the navigation method should be invariant or at least to some extent tolerant against such perceptual effects. Prerequisites (iii) and (iv) are closely related to the processing of the visual information, and a failure will result in imprecise, inaccurate, or —from the user's perspective— unexpected navigation behavior.

3.2.3.1. Robustness Against Perceptual Aliasing

Perceptual aliasing, sometimes also referred to as "spatial aliasing", occurs if identical sensor data is perceived at two different positions in space (reviews: [175, 432]). In the context of vision, different places with identical visual appearance frequently occur in repetitive environments. Without

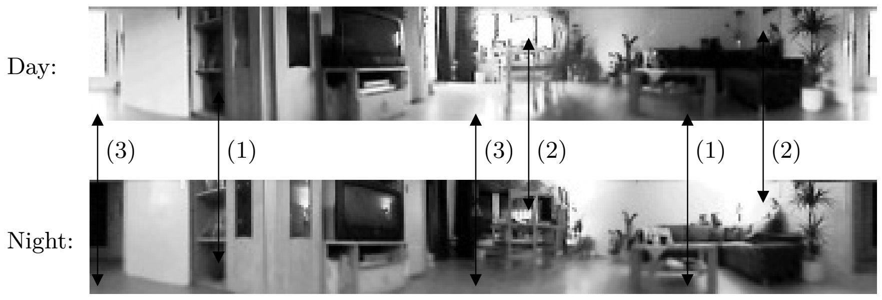


Figure 3.3.: Influences of changes of the illumination. The images were acquired at the same position in space but at different points in time and under different illumination conditions: under natural illumination during day (top) and under artificial illumination during night (bottom). The visual appearance of an image changes due to changes of the shadow cast (1), of image intensities (2), and of the positions of highlights and reflexions (3).

metrical position information, places with identical appearance cannot be disambiguated. Thus, qualitative methods (methods not incorporating position information; section 3.2.2) are more prone to perceptual aliasing than quantitative methods (relying on position information; section 3.2.2). Robustness against perceptual aliasing can also be achieved by considering the current sensor data together with sensor measurements recorded some short time ago (e.g. [327, 440]). This "history" helps to disambiguate places. For our particular application, robustness against perceptual aliasing is in particular important for detecting if the robot approaches an already cleaned area (loop-closure detection; section 3.2.3.3). A failed detection leads to repeated coverage or uncleaned areas.

3.2.3.2. Robustness Against Perceptual Variability

Perceptual variability occurs if the visual appearance of a place changes over time (reviews: [175, 432]). Robustness against perceptual variability is an essential aspect for every visual navigation method because *illumination changes* or *scene changes* can dramatically alter the visual appearance of an image. These two aspects will be further discussed in the following.

Illumination Changes

Changes of the illumination conditions influence the image intensities, the shadow casts and the positions of reflexions and highlights (figure 3.3). They can be categorized into three groups: *daytime changes*, *medium-term changes*, and *abrupt changes*. *Daytime changes* are illumination changes being due to the position of the sun changing over the entire day. They only influence robot navigation if the robot operates over a long period of time or if it resumes operating after some time, e.g. after recharging batteries. A typical cleaning robot has a battery capacity of approximately 1 h. In case it is capable of cleaning its workspace with one battery charge, we expect the influence of daytime changes to be rather low. If the robot has to resume its cleaning process after recharging, daytime changes are likely to influence the navigation behavior. *Medium-term changes* of the illumination are usually due to certain weather situations such as alternating clouds and sunshine. As they occur within several minutes, they can strongly influence a single cleaning run. *Abrupt changes* of the illumination can occur within seconds. They are in most cases caused by artificial changes of the illumination such as switching lights on or off. In rare cases, abrupt changes can be due to weather conditions. Such changes can also occur during a single cleaning run.

Robustness against changes of the illumination can be achieved at different processing steps of a navigation algorithm: (i) at image acquisition, (ii) at image preprocessing, and (iii) at the algorithmic level. The first level was considered in the diploma project by Dr. Sven Kreft [342] and the bachelor's project by Gereon Götze [243] both supervised by Lorenz Hillen. In both cases, a camera controller is applied to keep the average image brightness constant. The second level was addressed by Björn Böttcher in his bachelor's thesis [60] also supervised by Lorenz Hillen. There, different methods for preprocessing methods and image dissimilarity measures[1] were tested for local visual homing and visual localization. This approach is pursued for the holistic loop-closure detection methods described in chapter 5. The third level includes, for example, to develop homing algorithms or loop-closure detection methods being robust against changes of the illumination. For making our algorithms illumination-tolerant we currently focus on the used image dissimilarity functions. In the bachelor's thesis by Björn Böttcher [60] and in chapters 5 and 6 of this dissertation we therefore compare a wide range of image dissimilarity functions proposed in the literature (reviews: [19, 91, 92, 225, 635]). In addition, Prof. Dr. Ralf Möller recently developed new dissimilarity functions for application with local visual homing algorithms (Möller [448–450]).

Scene Changes

Perceptual variability can also be caused by changes of the visual environment. Such changes can be divided into *dynamic* and *static* scene changes. *Dynamic scene changes* are the most likely scene changes and are due to persons or pets moving in the robot's workspace. *Static scene changes* are caused by modifications of the environment such as moving, removing or adding objects or furniture. The influence of a scene change on navigation performance strongly depends on the extent of the disturbance in the perceived image. Smaller disturbances are more likely to be tolerated, whereas larger disturbances can cause the navigation method to fail. Due to the large field of view, most disturbances only affect a small portion of the entire omnidirectional image and therefore do not have a large influence on the used navigation strategy. Scene changes have to be considered at the algorithmic level. One possible approach would be to first detect disturbed image regions or erroneous correspondences and later on discard them in subsequent computations. Scene changes were not further considered within this thesis and static environments were assumed for our robot experiments (sections 4.4 and 6.4). Assuming a static environment for a cleaning robot is reasonable if the user is not at home or at least does not move in the room currently cleaned by the robot. It is clearly violated if people or animals are moving in the currently cleaned room, follow the robot, or even block its path.

3.2.3.3. Correct Place Recognition

The notion of a place is crucial for any navigation method. As discussed in section 3.2.1, navigation requires at least to recognize the robot's goal, but —depending on the robot's task— it can also involve to memorize and recognize several places. On the level of local visual homing, where the robot is supposed to return to a previously visited position, not correctly recognizing the goal can cause the robot either to stop at an erroneous goal position or to not find the goal at all. For map-based navigation, not correctly recognizing places already mapped leads to *inconsistent maps*. In this case, the map will not correctly represent the robot's environment e.g. because of mapping identical places multiple times (figure 3.4). This problem is referred to as loop-closure problem (reviews: [99, 432, 631]; textbook: [586, 630]). If loop closures are detected correctly, spatial relations can be corrected, and the quality of the resulting map can be considerably improved (section 3.6.1.2). We consider loop-closure detection to be one aspect of the more general problem of recognizing places and therefore favor the term "place recognition" rather than "loop-closure detection". The principles of using visual information to represent and recognize places is introduced in section 3.3. In the context of visual cleaning-robot control, reliable place recognition is essential to combine several segments of parallel and meandering lanes for completely covering complex-shaped workspaces. When approaching an already cleaned segment, the robot has to stop at the segment's

[1] Also referred to as "image distances". Throughout this dissertation we prefer the term "image dissimilarities" in order to emphasize the difference to *spatial* distances.

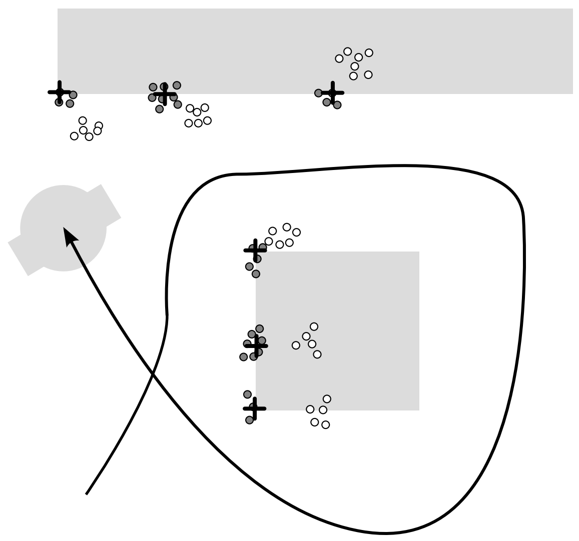


Figure 3.4.: Naive mapping method without sensor-data integration and place recognition. While driving along its trajectory (thick line), the robot continuously senses features (black crosses) and maps each measurement (circles). At the beginning, the position of the measurements (dark-gray filled circles) are close to the true feature positions (crosses). When the robot detects the previously sensed features again, the estimated feature positions (open circles) deviate from their true positions (crosses) resulting in inconsistencies. Consistent mapping is only possible if loops are correctly detected and the current sensor data is correctly integrated into the existing map.

border in order to avoid repeated coverage or gaps between neighboring segments. Chapters 5 and 6 describe two different approaches to loop-closure detection in the context of cleaning robots.

3.2.3.4. Correct Sensor-Data Integration

Map-based navigation methods accumulate sensor data over a longer period in time and over various positions in space (review: [74]). The process of fusing the robot's current sensor data into an already existing map is referred to as *sensor-data integration*. Correctly integrating sensor data is a prerequisite for all mapping methods, and if it fails, maps will become inconsistent. Correct sensor data integration is closely related to *localization*, i.e. the process of computing the robot's position in space (not necessarily in a metrical sense). Without knowing the robot's position, the map cannot be updated or extended correctly resulting in inaccurate and inconsistent maps. This principle gave rise to the research field of simultaneous localization and mapping (SLAM; section 3.6.1.2). Section 3.6 describes in more detail how spatial and temporal integration of sensor data is achieved for various types of maps; sections 3.6.3.2 and 4.2 point out how aspects of sensor-data integration are used for our work on dense topo-metric maps.

3.2.4. Omnidirectional Vision

The principles of navigation discussed so far in this chapter are mostly independent of the used sensory information and hold for arbitrary sensors. In this section, we briefly introduce fundamental properties of omnidirectional vision (section 3.2.4.1) and discuss aspects related to omnidirectional vision for controlling cleaning robots (section 3.2.4.2).

3.2.4.1. Omnidirectional Vision in a Nutshell

Omnidirectional cameras (review: [566], textbooks: [40, 131, 586]) are vision setups providing a considerably larger field of view than traditional cameras (figure 3.5). In the strict sense of the term *omni*-directional, such cameras can sense in *all* directions resulting in a complete *spherical view* ([72, 566]; figure 3.5.1). Nevertheless, the common usage of the term *omnidirectional camera* also includes cameras for obtaining a *hemispherical view* (figure 3.5.2) or a *panoramic view* (figure 3.5.3). While *hemispherical views* cover at least half of the entire viewing sphere, *panoramic views* have a field of view including at least one great circle of the entire sphere (i.e. the plane defined by the circle contains the sphere's center).

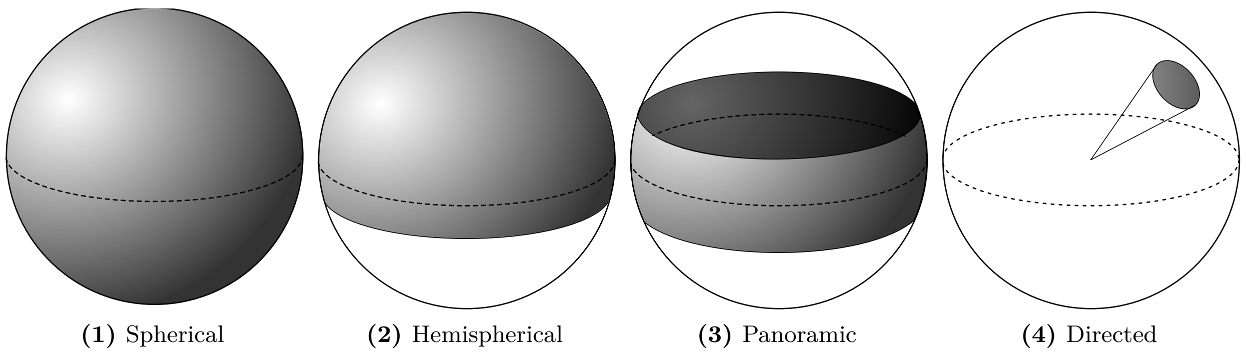


Figure 3.5.: Different camera types and their fields of view. The field of view decreases from left to right: subfigure (1): spherical views cover (almost) the entire viewing sphere, subfigure (2): hemispherical views image at least a hemisphere of it, subfigure (3): panoramic views image a subset containing a great circle of the entire sphere, and subfigure (4): directed views only image a small portion of it. According to its normal usage, the term *omnidirectional camera* subsumes cameras with spherical, hemispherical, and panoramic fields of view. Directed views are usually obtained by standard pinhole cameras. The dashed line depicts the horizon of the viewing sphere. Extended after Pajdla, Svoboda, and Hlaváč [498] and Bunschoten [72].

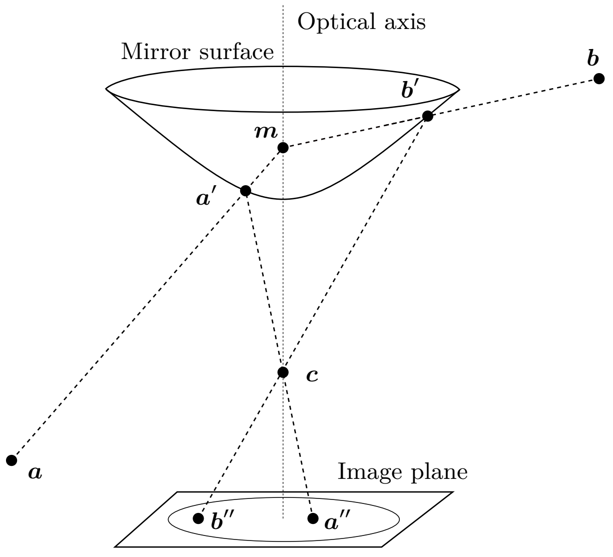
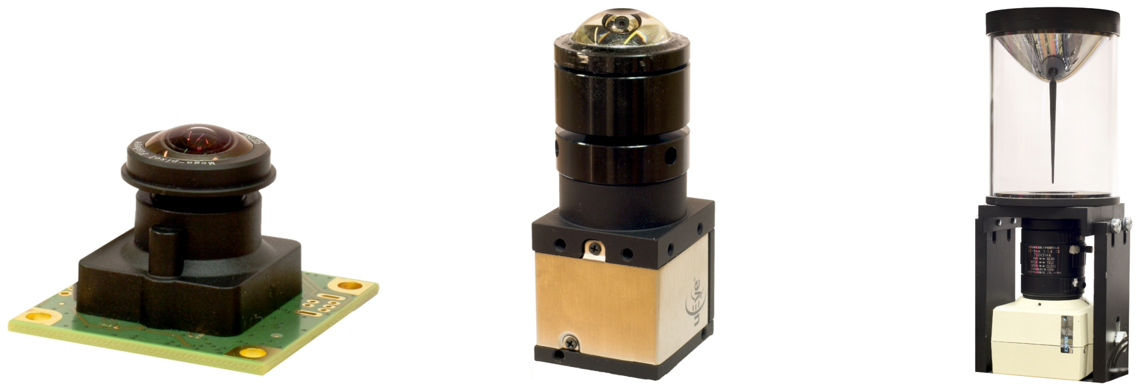


Figure 3.6.: Omnidirectional vision setup with a single projection center. Light (dashed lines) incident from scene points $\boldsymbol{a}$ and $\boldsymbol{b}$ intersects the mirror surface in points $\boldsymbol{a}'$ and $\boldsymbol{b}'$, respectively. In the depicted case, the incident rays both intersect in the point $\boldsymbol{m}$ inside the mirror surface, which is usually referred to as projection point or viewpoint. If this is the case for rays incident from arbitrary scene points, the setup is referred to as *single viewpoint* or *central*. For non-single viewpoint setups, incident rays do not intersect in a single point. Rather the area of intersection forms a surface (e.g. [613, 614]). The points $\boldsymbol{a}'$ and $\boldsymbol{b}'$ on the mirror surface are imaged to points $\boldsymbol{a}''$ and $\boldsymbol{b}''$ on the image plane. The rays of light from $\boldsymbol{a}'$ to $\boldsymbol{a}''$ and from $\boldsymbol{b}'$ to $\boldsymbol{b}''$ intersect in the camera's projection point $\boldsymbol{c}$. This is due to standard pinhole cameras also fulfilling the single-viewpoint constraint (textbook: [586]). After [I83].

Due to the image distortions resulting from their large field of view, traditional camera models for pinhole cameras (review: [603], textbooks: [186, 278, 641]) cannot be applied to omnidirectional vision setups. Hence, omnidirectional vision setups require particular camera models. A detailed mathematical description of such models is beyond the scope of this thesis; instead, the reader is referred to the original publications [26, 224, 564, 565] and the textbooks focusing on robotics [586] and on omnidirectional vision [40, 131]. Here, we rather focus on fundamental properties of such sensors.

Omnidirectional vision setups can be categorized into systems with *single viewpoint* or with *non-single viewpoint* (e.g. [40, 566, 586]). The former are also referred to as central systems. For *single viewpoint* setups, the incident rays of light all intersect in a single point referred to as projection point or viewpoint (point $\boldsymbol{m}$ in figure 3.6; [566]). Every pixel in the sensed image then corresponds to exactly one ray of light passing through the viewpoint. Single-viewpoint setups facilitate camera calibration, image unfolding, and the mathematical models required for describing such setups [40, 224, 564–566]. In many cases, *non-single viewpoint* setups can be treated as having an approximate single viewpoint. For further details on non-single viewpoint setups, the reader is referred to Swaminathan, Grossberg, and Nayar [613].



(1) Dioptric setup with fisheye lens. H: 2.5 cm, ∅: 2 cm, FoV: -2.5°/90°. Camera: IDS uEye UI-1246-M [D7], lens: Sunex DSL215 [D14].

(2) Catadioptric setup with panoramic annular lens. H: 8 cm, ∅: 3 cm, FoV: -17°/38°. Camera: IDS uEye UI-2220-M [D8], PAL: Tateyama PAL-S25G3817-27C [D16].

(3) Catadioptric setup with hyperbolic mirror. H: 20 cm, ∅: 7 cm, FoV: -90°/46°. Camera: ImagingSource DFK4303 [D11], lens: Pentax TS2V314A [D5], mirror: Accowle Large Type Wide Angle [D1].

Figure 3.7.: Examples of omnidirectional vision setups used at the Bielefeld Computer Engineering group. Please note that the setups depicted in subfigures (1) to (3) are reproduced at different scales. The used abbreviations are: H: height, ∅: diameter, FoV: field of view with the first and the second angle denoting elevation below and above the horizon. Photos by Lorenz Hillen. Figure best viewed in color.

For acquiring omnidirectional images, different types of sensors can be used (see figure 3.7 for examples). According to Scaramuzza [566], omnidirectional vision setups can be divided into *dioptric setups*, *catadioptric setups*, and *polydioptric setups*. *Dioptric setups* are a combination of a camera and a fisheye lense with a field of view larger than 180°. To this end, all dioptric sensors generate hemispherical views (figure 3.5.2). Recent fisheye lenses are developed to have a single viewpoint. Due to imaging an entire hemisphere, the vertical field of view above the horizon equals 90°. However, it does not exceed much below the horizon. The angular resolution of dioptric setups is not constant over their field of view but decreases from the center to the outer border of the resulting camera image. As the horizon is imaged at the outer image border, it is imaged with the setups lowest (i.e. worst) resolution. Furthermore, fisheye lenses are prone to vignetting, i.e. the darkening towards the borders of the image. These two drawbacks can limit the applicability of dioptric setups for generating panoramic views. Due to the advances in camera and lens miniaturization, dioptric setups are the smallest omnidirectional vision setups we are currently aware of. For this reason, our prototype cleaning robot (section 4.4.3) was equipped with such a setup (figure 3.7.1) after the robot experiments presented in this thesis (sections 4.4 and 6.4) were conducted.

Catadioptric setups are the most widely used setups for acquiring omnidirectional images. They combine a standard camera with a mirror surface being rotationally symmetric around the camera's optical axis. Common mirror surfaces are hyperbolic, elliptical, or parabolic. It was proven by Baker and Nayar [26] that hyperbolic and elliptical mirror shapes satisfy the single-viewpoint constraint in combination with standard lenses; parabolic mirrors only fulfill the constraint in combination with a telecentric lens (i.e. a lens capturing only incident rays parallel to its optical axes; textbook: [600]). Conical or spherical surfaces are possible but do not fulfill this constraint [26, 40]. Most catadioptric setups can be used for generating hemispherical views and offer —due to the camera facing the mirror (figure 3.7.3)— offer a 90° field of view below the horizon. (exceptions are mirror without smooth tip). The field of view above the horizon is usually large, and it's extent is strongly influenced by the shape of the mirror surface. Catadioptric setups can be designed to have constant angular resolution [90, 605, 607]. Most robotic applications rely on catadioptric setups for generating panoramic views. Traditional catadioptric setups are usually very large (both in the diameter and the height of the setup; figure 3.7.3). It is only recently that small mirrors became available for mobile phones [I41, I44, I62] and consumer digital cameras [I90]. At the current state, these setups cost approximately 20 €, and we are not aware of an application of such mirrors for robot navigation. Between the two extremes, catadioptric sensors of compact size can be built by integrating the mirror directly into the camera's lens. Examples of such sensors include the panoramic annular lens (PAL, [363, 517], patent: Greguss [251]; figure 3.7.2) used for the robot experiments of this thesis (sections 4.4 and 6.4) and the mirror-lens combinations proposed by Stürzl et al. [605], Stürzl and Srinivasan [606], and Stürzl, Suppa, and Burschka [607].

Polydioptric setups consist of several pinhole cameras with overlapping fields of view. Such setups are also referred to as *multi-camera rigs*, and the PointGray Ladybug [I43] is the most well-kown example of such setups. Depending on the number of cameras and their arrangement, polydioptric setups can acquire spherical views, hemispherical views, and panoramic views. Using several cameras to acquire an omnidirectional image is both the main advantage and the main drawback of these setups. On the one hand, they allow for creating images with a large resolution (e.g. Google StreetView relies on such setups; Anguelov et al. [10]). On the other hand, the cameras cannot be arranged such that the entire setup fulfills the single-viewpoint constraint. This will always induce motion parallax, especially if the robot is moving close to obstacles [85, 318].

All these sensors are *monocular* sensors. Like monocular pinhole cameras, they suffer from the limitation that it is not possible to estimate the distance of an imaged feature to the camera (i.e. its depth) from only one image. The depth can be computed by exploiting the two-view geometry (pinhole cameras: [278], omnidirectional cameras: [291, 498]). This requires either two monocular images taken at different positions in space or an omnidirectional stereo setup. Omnidirectional stereo setups provide two different omnidirectional images from the same position in space. This can be achieved by stacking two monocular catadioptric sensors above each other [234, 484] or by using a single camera facing two different mirror surfaces [84, 307, 423, 425, 608, 610]. As the methods described in this thesis all rely on monocular images, we do not further discuss omnidirectional stereo. Although omnidirectional stereo vision would allow to compute depth information from two images acquired at the same position in space (section 3.3.2), we see the following drawbacks: (i) omnidirectional stereo setups are more complex and hence also more expensive than monocular setups, (ii) if two camereas are stacked (as depicted in figure 3.18.1), two monocular setups are needed, and (iii) the computational effort of the image processing methods is larger than for processing monocular images.

Some navigation strategies relying on omnidirectional images operate on the *camera images* (e.g. [7, 8, 206, 235, 237, 292, 364, 472, 476, 477, 561, 568, 619, 653]; figure 3.8 left), whereas most operate on the *unfolded images* (e.g. [5, 6, 14, 54–56, 655]; figure 3.8 right). It strongly depends on the application, whether camera images or unfolded images are better suited. Operating on *camera images* is suitable for applications using the image regions around the poles of the viewing sphere (e.g. for observing the floor or the sky) (i) because these regions cover a large portion of the image area and (ii) because these these regions would appear strongly distorted in unfolded images. In the camera images, vertical lines in the environment are imaged as radial lines intersecting at the principal point, i.e. the intersection between the camera's optical axis and the sensor plane. Horizontal lines, i.e. lines with identical elevation (including the horizon with zero elevation), are imaged as circles centered at the principal point. Furthermore, the resulting camera image depends on the geometry and properties of the used omnidirectional vision setup.

We consider operating on *unfolded images* to be more appropriate if the developed algorithms are supposed to be used with different omnidirectional vision setups. Unfolding usually includes a mapping to spherical or cylindrical coordinates [40, 131]. By this means, identical panoramic images can be generated independent of the particular omnidirectional vision setup. Thus, applying

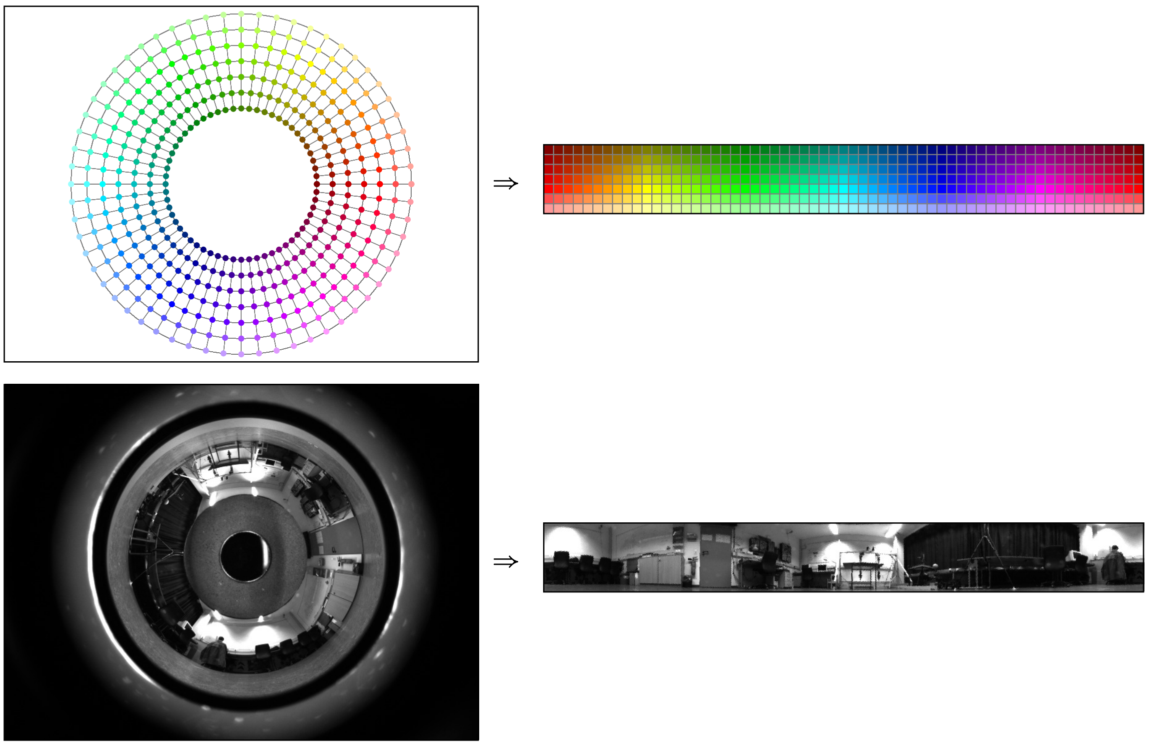


Figure 3.8.: Unfolding omnidirectional images. By an appropriate mapping, the camera images (left) are transformed to panoramic images (right). Top: schematic drawing; bottom: real images obtained by one of our omnidirectional vision setups (camera: IDS Imaging UEye UI-2220SE-M [D8]; panoramic annular lens: Tateyama PAL-S25G3817-27C [D16]). Figure requires color printing.

algorithms developed for unfolded images with different omnidirectional camera setups only requires the application of the appropriate unfolding method, but no further adjustments. Although unfolded images contain the same image information than camera images, unfolded images resemble more how we humans see the world: vertical and horizontal lines in the scene are imaged as vertical and horizontal lines, respectively. This property facilitates visual inspection of onmidirectional images, especially if the panoramas contain the region around the horizon, which appears in the camera images curved and compressed to a small portion of the entire image. All navigation strategies presented in this thesis rely on unfolded images.

3.2.4.2. Omnidirectional Vision for Control of Cleaning Robots[2]

The goal of the navigation strategies presented in this thesis is to make an autonomous floor-cleaning robot capable of systematically and completely covering its workspace. This suggests using more elaborated sensors like those used by other second-generation robots. Such sensors include directed cameras or laser-range finders (section 2.2.1). Although omnidirectional vision is currently not used by comparable commercial or academic floor-cleaning robots, we think it is an appropriate sensor for this task. In our opinion, the five main advantages of using such an omnidirectional-camera setup include (i) that the robot is capable of capturing a panoramic view of its environment with only a single camera image, (ii) that it mainly perceives the region around the horizon, (iii) that the visible image content is independent of the robot's orientation, (iv) that it facilitates ego-motion estimation, (v) that dense sensory information is obtained, and (vi) that —regarding a potential product— this working direction is not blocked by patents. The aspects (i) to (v) will be discussed in the following.

[2]This section is an extension of section 3.3.1 in our journal publication Gerstmayr-Hillen et al. [222]

Taking a panoramic view with a single camera image requires the application of a dioptric or a catadioptric sensor. As the local visual homing algorithms operate on low-resolution images (section 3.5.2), we do not need the large image resolutions which can be achieved by polydioptric sensors. Furthermore, such setups are more expensive as they require more than one image sensor and additional lenses. Regarding our particular application, the main drawback of dioptric or catadioptric setups is that they cannot be completely enclosed by the robot's housing and therefore protrude. Thus, a cleaning robot could damage its sensor while cleaning underneath furniture such as beds or cupboards. This effect can be reduced by relying on compact catadioptric setups like panoramic annular lenses as used for our experiments (figure 3.7.2) or small-sized dioptric setups (figure 3.7.1). However, even such sensors cannot be completely enclosed by the robot's housing as is the case for monocular cameras directed upwards.

Compared to a standard camera directed upwards, an omnidirectional vision setup mainly perceives the region of the horizon which often contains (i) more visible structure than the ceiling and (ii) the obstacles which can block the robot's way. Because of their 360° field of view and their viewing direction, visible objects do not vanish from panoramic images due to robot movements but only due to environmental properties such as occlusions. Furthermore, when the robot is moving back and forth along meandering lanes, the same objects are visible independent of the direction of travel. We think that this is an important advantage over standard cameras. However, these are exactly the properties making omnidirectional methods more prone against image disturbances or perceptual variability (section 3.2.3.2). Due to their large field of view, the chance that perceptual variability due to dynamic scene changes or changes of the illumination influence the image is larger, but their influence on the navigation method depends on the portion of the affected image area. This is in contrast to standard cameras facing the ceiling, especially if these cameras only have a narrow field of view. In this case, even small changes affect the entire image and therefore have a large impact on the navigation capabilities.

For the visible image content being independent of the robot's orientation in the plane, the robot's axis of rotation needs to be identical to the optical axis of the vision system. If this is the case, a change of the robot's orientation can be simulated by horizontally shifting the columns of the panoramic image. In the context of this thesis, this property is exploited for loop-closure detection (section 3.3 and chapters 5 and 6), i.e. for detecting if the robot approaches an already visited position.

Omnidirectional images facilitate ego-motion estimation because they allow for easily separating rotational and translational movement components (Nelson and Aloimonos [486]). Camera motions give rise to characteristic changes of the visible image content. For translational movements, two image regions exist for which the visible image features are only expanded or contracted but not shifted. These points are referred to as *focus of expansion* and *focus of contraction* in terms of optical flow literature (textbooks: [305, 641], reviews: [30, 280]) and as *epipoles* in terms of stereo literature (textbooks: [278, 641]). Rotations of the robot give rise to a constant shift of image features independent of the feature's distance to the camera. In case of *small* combined movements, the two types of changes are superimposed, but the foci of expansion and contraction persist.[3] For movements in the plane, the foci of expansion and contraction are always visible in panoramic images with their 360° field of view, but they may not be visible in directed camera images with their limited field of view. For this reason, omnidirectional images allow for better separation of rotational and translational components.

Images are usually represented as a two-dimensional pixel grid of color or intensity values. Thus, images provide dense three-dimensional information about the robot's environment. This aspect distinguishes vision from other sensor modalities. Most laser-range finders sense a dense one-dimensional pixel grid with distance and reflectance information, and IR distance sensors only

[3]This property does not longer hold if sufficiently *large* rotational motion components.

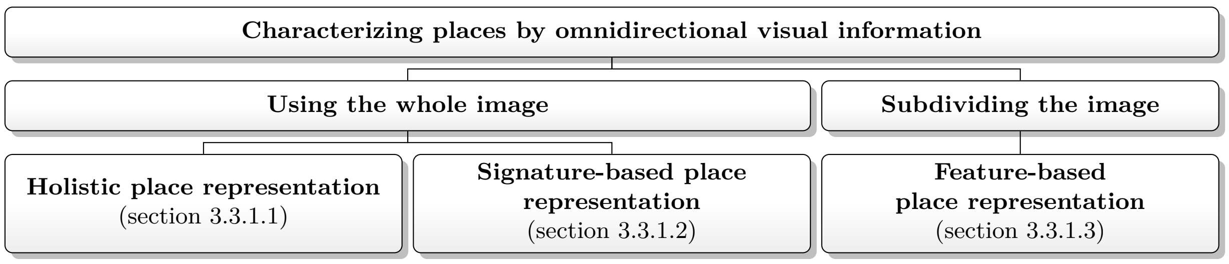

Characterizing places by omnidirectional visual information
Using the whole image
Subdividing the image
Holistic place representation (section 3.3.1.1)
Signature-based place representation (section 3.3.1.2)
Feature-based place representation (section 3.3.1.3)

Figure 3.9.: Characterizing places based on omnidirectional visual information. The grouping depends on how the visual information is used to represent places.

return sparse direction and distance information. Furthermore, the sensory range of such sensors is limited, and their uncertainty increases with increasing measurement distance. Due to their dense sensory information, both omnidirectional and directed images cannot only be used for mapping and estimating spatial relations but also for other tasks. Such tasks include obstacle detection and mapping (e.g. by time-to-contact methods [321, 418, 419] or based on optical flow [302, 670]) or user interaction based on gesture recognition [98, 405, 573].

3.3. Representing and Recognizing Places

Places form the lowest level of the hierarchy of navigation strategies proposed in section 3.2.2. Based on the omnidirectional visual information, vision-based navigation methods characterize places, navigate between places, determine spatial relations of places, or integrate several places into a map. In the following, we first describe how omnidirectional visual information can be used to characterize and recognize places (section 3.3.1). Recognizing an already visited place is essential for nearly all navigation capabilities discussed in sections 3.4 to 3.6 and involves comparing the representations of two or more places.

For comparing place representations, a dissimilarity value is computed with zero dissimilarity typically expressing identical visual information and hence identical places in space. Throughout this thesis, we prefer the term “image dissimilarity” over the commonly used terms “image distance” or “image difference” in order to emphasize the difference between *image* dissimilarity and the *spatial* distance between the positions of image acquisition. The computed dissimilarity measure can be used (i) for a binary decision whether or not the compared visual information is identical or (ii) for images acquired spatially close to each other as a measure of the spatial distance between positions of image acquisition (Zeil, Hoffmann, and Chahl [718]). Beyond that, many visual navigation methods relate the perceived visual information with a unique metrical position in space, i.e. with a coordinate. By this means, a *qualitative* navigation strategy can be turned into a *quantitative* one. Techniques for position estimation based on visual information are introduced in section 3.3.2.

3.3.1. Representations of Places Based on Omnidirectional Visual Information

Depending on how the image is used to characterize or represent a place, the following three approaches can be distinguished (figure 3.9): (i) *holistic place representation* (section 3.3.1.1), (ii) *signature-based place representation* (section 3.3.1.2), and (iii) *feature-based place representation* (section 3.3.1.3). A similar classification is also proposed by Gonzalez-Barbosa and Lacroix [242]: depending on how places are characterized, the authors label the categories *global attributes*, *space transformations*, and *local attributes*, respectively.

Figure 3.10.: Holistic place representation. The omnidirectional image $\boldsymbol{I}$ itself or a preprocessed variant $\boldsymbol{P} = \mathrm{p}(\boldsymbol{I})$ is used to characterize places.

3.3.1.1. Holistic Place Representation

Holistic representations of places use the entire omnidirectional image to characterize a place (figure 3.10). This can either be the raw intensity information (as depicted in figure 3.10) or a preprocessed image. Typical preprocessing techniques used for holistic place representations with omnidirectional images include local contrast normalization ([441, 612, 698, 720] and chapter 5) or edge detection (chapter 5). A holistic representation of places saves the computational effort (i) of detecting local image features and computing feature descriptors as it is required by feature-based approaches (section 3.3.1.3) and (ii) of transforming the image information into a global image signature as is the case for signature-based approaches (section 3.3.1.2). However, it requires to store the entire image or, in case of map-based navigation, of several images. The storage capacity depends on the size of the images and the number of images which have to be stored to accomplish the robot's task.

For holistic representations of places, recognizing a known place involves comparing the known and the current place representation by applying an image dissimilarity function (reviews: [19, 91, 92, 225, 635]). Image dissimilarity functions compute the dissimilarity of the considered images in a two step process. First, they compare the images pixel-by-pixel by correlation techniques or by computing intensity differences. In the second step, these pixel-wise dissimilarities are fused to an overall estimate of the image dissimilarity. This is usually accomplished by summing or averaging over the values computed for each pixel. Frequently used functions include the sum of squared differences (SSD) or the normalized cross correlation (NCC). The compared images need to be aligned w.r.t. a common reference direction. This can be achieved by applying a visual compass method (section 3.4) and accordingly rotating one of the images to compensate for the compass shift. The computational effort required for comparing or recognizing places strongly depends on the resolution of the used images. Typical navigation strategies relying on holistic representations of places use a relatively low resolution of approximately 1° per pixel [447, 451, 453, 456]. This low resolution makes navigation methods operating on a holistic representation of places also suitable for robots with limited computational power and storage capacities.

Without choosing appropriate building blocks, navigation strategies relying on a holistic representation of places are prone to perceptual variability (section 3.2.3.2). The changes of the visual appearance resulting from dynamic scene changes or illumination changes can strongly decrease the navigation performance of holistic methods because they use the entire and often unprocessed image to represent a place. Robustness against changes of the illumination can be achieved by choosing (i) appropriate preprocessing methods or (ii) illumination-tolerant dissimilarity functions. In the ideal case, the combination of both would achieve true invariance: independent of the conditions under which the compared images were acquired, preprocessing methods would completely eliminate the influence of image disturbances and image dissimilarities functions would compute an identical dissimilarity value. More realistically, currently used techniques can only tolerate such image changes up to a certain extent, and navigation is likely to fail for stronger changes influencing large portions of the visible image information. Robustness against dynamic scene changes is more difficult to achieve for navigation strategies operating on a holistic place representation. If the scene changes result in small changes of the visual appearance, it is likely that they are tolerated by the navigation strategy. Such changes are usually restricted to a small and local image area (e.g. resulting from

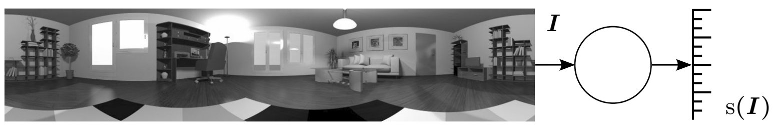


Figure 3.11.: Signature-based place representation. Places are characterized by low-dimensional global signatures $s = \mathrm{s}(\boldsymbol{I})$ derived from the entire omnidirectional image $\boldsymbol{I}$.

adding or removing an object in the robot's environment). For changes influencing a large portion of the omnidirectional image, navigation is likely to fail.

Holistic representations of places are used with *holistic compass methods* (section 3.4.2.1), *topometric maps* (section 3.6.3.2), and *purely topological maps* (section 3.6.4.1); homing methods relying on a holistic representation of places include *warping methods*, *DID methods*, *local optical flow methods* and *correspondence-based methods without feature preselection* (sections 3.5.2.2 and 3.5.2.3).

3.3.1.2. Signature-Based Place Representation

Signature-based approaches derive a global image signature from the entire omnidirectional image and use this signature to represent the place (figure 3.11). In some cases, the signature is also referred to as fingerprint [621–623, 687, 688] or as global image descriptor [242]. As the signature computed from the image information has a much lower dimensionality than the image itself, signature-based approaches allow for very efficient image comparisons and require only a small amount of storage. The additional overhead of deriving the signature by evaluating a *signature function* to compute the signature is usually negligible. These properties make them interesting for developing navigation strategies for robots with limited computational power. Signature-based representations of places are strongly influenced from image retrieval (review: [132]). There, a single image needs to be efficiently compared to a potentially large number of stored reference images. In the context of appearance-based robot navigation, the currently perceived image is compared to a set of images stored in the robot's map.

Global image signatures as a low-dimensional representation of omnidirectional images can be partitioned into *rotation-invariant* and *rotation-dependent* signatures (figure 3.12). Hence, signatures computed from images acquired at identical position in space but different orientation of the robot are identical (rotation-invariant signatures) or differ (rotation-dependent signatures). In some cases (e.g. Fourier signatures or statistical signatures), the same mathematical principles can be used to derive rotation-dependent and rotation invariant signatures. Whether or not a signature is rotation-invariant then depends on the exact computation of the signature (figure 3.13). In the following, we will briefly describe the most important groups of signatures.

Fourier signatures are obtained by a Fourier transformation (textbook: [65]) of the omnidirectional image. As omnidirectional images are periodic in horizontal but not in vertical direction, images are usually transformed row-by-row with a one-dimensional Fourier transformation. The signature is then formed by the low-frequency Fourier components which cover the relevant image content. The information contained the higher-frequency components (i.e. noise) is discarded. The resulting signature is rotationally invariant if the phase information is removed (e.g. by computing absolute Fourier coefficients), whereas it is rotation-dependent if the phase information is kept.

Statistical signatures use statistical measures such as the average image intensity or the average color value to describe the image. Depending on the image region for which the statistical measures are computed, the resulting signatures are rotationally invariant or not. Rotation invariant signatures are obtained by deriving the signature from the entire image or from computing the statistical measure row-by-row (figure 3.13.1), whereas rotation-dependent signatures are obtained by column-wise computations (figure 3.13.2). Based on the same principle, rotation-invariant and rotation-dependent *histogram-based signatures* can be computed, which describe the image by an intensity histogram

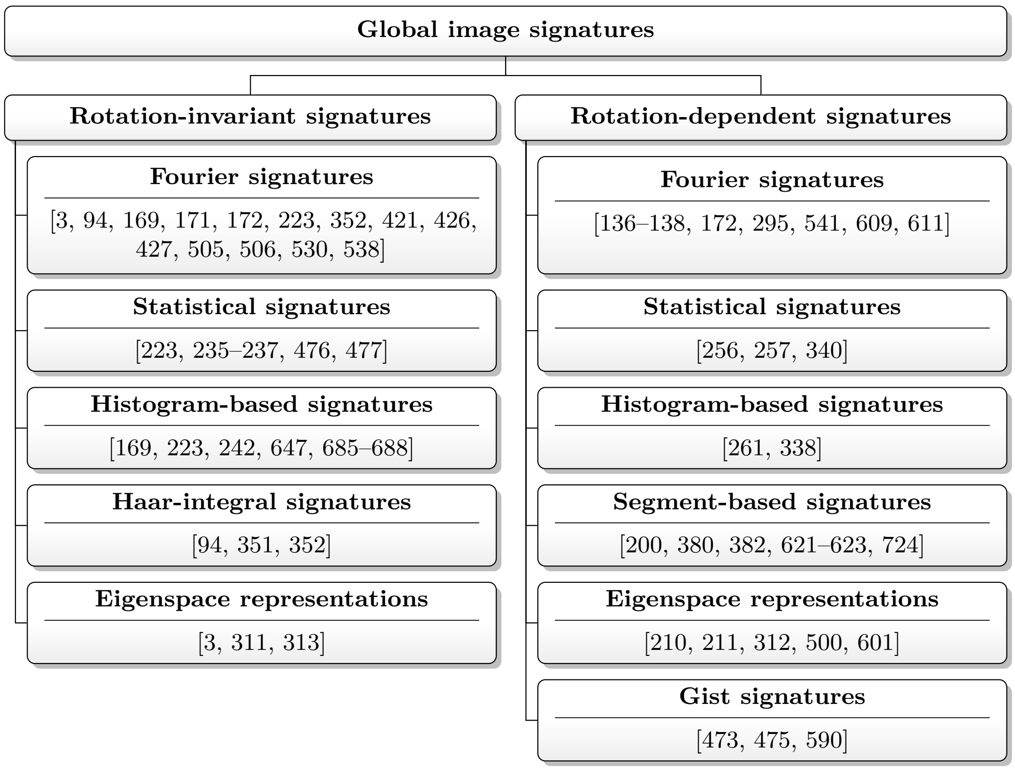
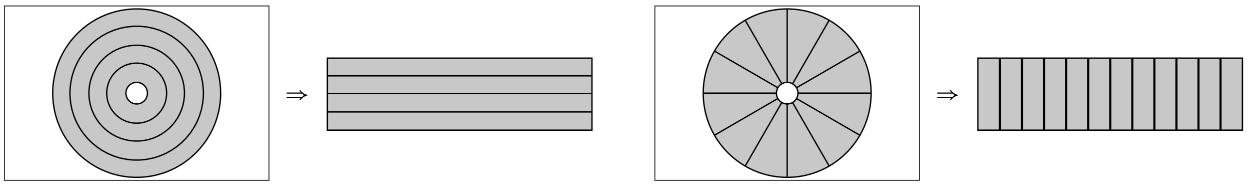


Figure 3.12.: Categorization of global image signatures.



(1) Rotation-invariant signatures. *Row-wise* computations result in descriptors which are independent of the robot's orientation.

(2) Rotation-dependent signatures. *Column-wise* computations yield global image signatures which depend on the robot's orientation.

Figure 3.13.: Computation of rotation-invariant (subfigure (1)) and rotation-dependent (subfigure (2)) global image signatures. The terms *row-wise* and *column-wise* refer to the *cylindrical* image depicted at the right of each subfigure.

[169, 223], a color histogram [338, 647, 687, 688], or an edge histogram [261].

Haar integrals (Siggelkow and Burkhardt [587]) are widely used in image retrieval. The *Haar-integral signatures* applied by [94, 351, 352] are specifically tailored to omnidirectional images and offer invariance against rotations and against certain types of illumination changes.

Segment-based signatures subdivide the image into a set of segments, which are in most cases represented by low-level features such as color blobs or vertical lines. Each segment is typically represented by a number or a letter representing the segment (e.g. "R" for red color blobs or "V" for vertical lines). All the representatives of an image are combined to a list which is used as signature. In contrast to all other reviewed signatures, the dimensionality of these descriptors varies depending of the number of segments detected in the image. Furthermore, the descriptors depend on the robot's orientation, but the ordering of list entries is independent of the robot's orientation. By comparing the ordering of the segment representatives (rather than their positions within the lists), rotationally invariant comparison functions can be obtained. In the works by [200, 724], the signature is built from a small number of outstanding colors. [380, 382] detect vertical lines and use image regions between neighboring lines as segments, which are represented by the average color value. The signatures proposed by [621–623] do not only contain low-level color or edge information, but also use doors and corners detected in a laser-range scan as segments.

Global image signatures based on *eigenspace representations* involve a principal component analysis (PCA, textbooks: [46, 533]) of the omnidirectional images. The low-dimensionality of eigenspace signatures results from using only the dimensions corresponding to the largest eigenvalues because these dimensions represent the most important visual information. The drawback of eigenspace signatures is that the transformation to the low-dimensional signature has to be be computed from a set of training images prior to robot navigation. Typically, eigenspace signatures depend on the robot's orientation (e.g. [210, 211, 312, 500, 601]), and the eigenspace representation is learned from a set of images acquired under constant robot orientation. Only the eigenspace signature proposed by [311, 313] offers rotational invariance and is learned from a set of images containing each image in several orientations. The training set of rotated images is obtained by step-by-step shifting omnidirectional images in horizontal direction.

Gist signatures[4] have to be considered a special case of rotation-dependent eigenspace representations. The signature was originally used to model human scene perception and recognition [492, 493] and is applied to represent and recognize places based on omnidirectional visual information by [473, 475, 590]. Computing the signature is a two-step process: (i) a high-dimensional descriptor is computed from processing the image with a bank of filters such as edge detectors or color detectors, and (ii) the high-dimensional descriptor is compressed by PCA.

Place recognition based on global image signatures requires comparing the signature of the currently perceived image with the stored signature representing the goal position or with several signatures stored in a map. For comparing *rotation-invariant* signatures, standard dissimilarity functions also applied for holistic representations of places (reviews: [19, 91, 92, 225, 635]) or comparison functions tailored to a certain type of signatures such as histogram comparison functions (reviews: [508, 549]) can be applied. Comparing *rotation-dependent signatures* requires either (i) aligned signatures or (ii) rotation-tolerant comparison functions. In the former case, the signatures have to be aligned w.r.t. a common reference direction before comparing them, e.g. by applying a signature-based compass method (section 3.4.2.2). Aligned signatures can be compared by applying standard dissimilarity functions. The latter case requires special comparison functions which compute a rotation-invariant dissimilarity measure although being applied to rotation-dependent signatures. In this casep, rotational invariance is in this case achieved by the combination of signature and comparison function.

For signature-based place representations, tolerance against illumination changes and dynamic scene changes can be obtained by computing or comparing signatures with functions which are tolerant against such changes. Like for holistic representations of places, signature-based place representations can only tolerate image disturbances up to a certain extent and will fail for stronger disturbances.

Signature-based place representations are used for *signature-based compasses* (section 3.4.2.2), for *parameter-based homing methods* (section 3.5.2.2), and for building *topo-metric* or *purely topological maps* (sections 3.6.3.2 and 3.6.4.1).

3.3.1.3. Feature-Based Place Representation

Feature-based representations characterize places by a set of local image features detected in the omnidirectional image (figure 3.14). Local image features are image patterns differing considerably from surrounding image regions (reviews: [226, 372, 438, 439, 577, 644], textbook: [586]). Such features include, among others, colored regions, textured regions, edges, or corners. They are in

[4]The term "gist" for this signature is not an acronym but was chosen because the signature covers the essential properties of a visual scene, i.e. its gist [492, 493, 585].

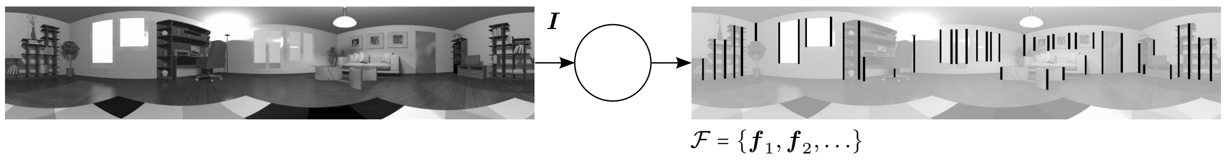

I
$\mathcal{F} = \{\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots\}$

Figure 3.14.: Feature-based place representation. The place is characterized by a set $\mathcal{F} = \{\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots\}$ of visible features (in the example vertical lines) each described by a feature descriptor $\boldsymbol{f}_i$. Features were manually selected; the panoramic image on the right is shown solely to visualize the feature set $\mathcal{F}$.

a first step extracted from the image information by a *feature detector*. Feature detectors are in the literature also referred to as *point of interest detectors* or *key-point detectors*. In a second step, regions around the selected points of interest are transformed to a *local feature descriptor*. Feature-based approaches then operate on the feature descriptors rather than on the intensity information. Depending on the used approach, detecting feature points and computing feature descriptors can be computationally demanding. However, if appropriate detectors and descriptors are chosen, features exhibit several properties making them interesting for robot vision (review: [644], textbook: [586]): (i) feature detectors are designed to reliably and repeatedly detect the same features even under different conditions of image acquisition such as a different illumination conditions or a different viewing directions, (ii) feature descriptors can exhibit invariance against various image changes including distortions, rotations, scale changes, or illumination changes, (iii) the computed descriptors are unique, i.e. the same image-intensity pattern will be transformed to the same descriptor, and (iv) the computed descriptors are distinctive, i.e. different intensity patterns result in different descriptors. Because of aspects (i) and (ii), feature-based approaches usually exhibit a good robustness against perceptual variability (section 3.2.3.2).

Feature-based representations of places are currently the standard approach in computer and robot vision, and the vast majority of omnidirectional visual navigation methods reviewed in this chapter rely on such a representation of places. Existing feature detectors and descriptors can be categorized into *corner detectors*, *line detectors*, and *blob detectors* ([586, 644]; figure 3.15). *Corner detectors* use intersection points of lines or edges as features. *Line detectors* rely on lines or edges as characteristic image features. In the context of omnidirectional vision, features are often restricted to lines resulting from vertical structures in the robot's environment. Such structures can be easily detected because they are imaged as radial lines in case the original camera image is used (figure 3.8 left) or as vertical lines in case the omnidirectional camera image is unfolded to a panoramic image (figure 3.8 right). *Blob detectors* find image regions (in contrast to points or lines) which differ from the surrounding image regions in intensity, color, or texture. Examples of local image features and omnidirectional visual navigation strategies relying on these features are given in figure 3.15.

To characterize an image based on the detected features, it is straightforward to store a list of feature descriptors. This representation is also referred to as *bag of features* and solely depends on the visible features but not on their position in the image (textbook: [586]). Recognizing a place then requires to match the features visible in the current image with the ones visible in the stored reference image, i.e. to establish correspondences. Therefore, pairwise dissimilarity values between feature descriptors are computed (in most cases by applying a standard norm function), and the best-matching pairs are identified as correspondences based on these dissimilarity values. The matching process is in principle a nearest neighbor search in the often high-dimensional feature-descriptor space. Its computational effort depends on (i) the number of features to compare and (ii) their dimensionality. As common descriptors like SIFT or SURF are 128-dimensional, only a small number of images can be compared for real-time control of a mobile robot. This fact in particular limits the scalability of map-based navigation methods relying on a feature-based place representation (section 3.6).

To speedup the process of establishing correspondences, efficient matching techniques operating

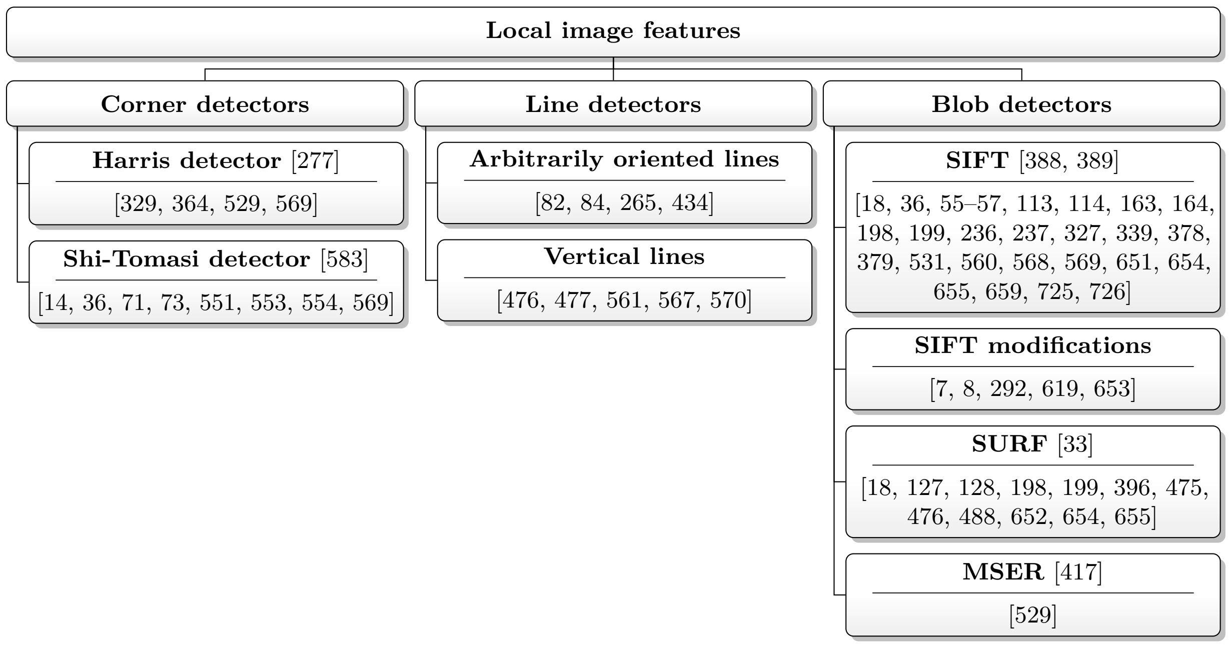


Figure 3.15.: Overview of local image features used with omnidirectional visual navigation strategies for wheeled mobile robots. For each detector, different references are given: references above the horizontal line point to the original publication; references below the this line refer to examples using the corresponding feature for omnidirectional visual navigation. Please note that the given lists of examples are not exhaustive. The used acronyms are: SIFT: scale invariant feature transform; SURF: speeded up robust features; MSER: maximally stable extremal regions. The categorization was adapted after Siegwart, Nourbakhsh, and Scaramuzza [586] and Tuytelaars and Mikolajczyk [644].

on image features were proposed. The *bag of words* method relies on a vector quantization of the descriptors (examples: [34, 127, 128, 394–396, 488, 531, 568], review: [130], textbook: [586]). It partitions the feature space into a set of cells. Each cell represented by a codebook descriptor and assigns each feature descriptor the most similar codebook vector. The vector quantization has to be learned prior to using it for navigation and is usually computed by clustering (textbooks: [46, 148, 533]) a set of feature descriptors detected in a set of training images. The resulting cluster centroids are then used as codebook vectors. Assigning an observed feature descriptor to its most similar codebook vector requires a nearest neighbor search in the high-dimensional descriptor space, which can in this particular case be performed more efficiently by a hierarchical and tree-based search process (examples: [127, 128, 488, 568], review: [130]). The search tree used for this purpose is also referred to as *vocabulary tree*. For representing places, each codebook vector is assigned a unique integer value (also referred to as *visual word* or *vocabulary*), and places are described by a set of one-dimensional integers rather than by a set of high-dimensional feature descriptors. This representation allows to efficiently compute the number of common features by comparing the two lists of visual words. This step can be implemented efficiently by an *inverted file* (examples: [34, 568], textbook: [586]). For each visual word, an inverted file lists which images contain the corresponding visual word. In order to compare the robot's current image with a set of known images, these lists are consulted for each feature of the robot's current image. The most similar image among the images stored in the map is the image sharing the largest number of common features with the current image.

With the feature-based matching methods described so far, images are considered to be identical if they share identical visible features. Depending on the properties of the robot's environment, the region for which the same features are visible can be relatively large. For more accurate place recognition, the results obtained by *bag of features* or *bag of words* methods have to be refined.

This can be accomplished by a *consistency check* or by *constrained matching* (review: [191]). Both approaches also contribute to make feature-based navigation strategies robust against perceptual variability because they discard ambiguous or erroneous matches which can occur because of illumination or scene changes.

Consistency checks detect and discard ambiguous matches by analyzing pairwise dissimilarities. For assessing the consistency of matches, two different approaches are currently used. The first one discards correspondences for which the dissimilarity value of the best match (i.e. of the correspondence) is not considerably smaller than the value of the second best match (smaller dissimilarity means a better match; e.g. [5, 6, 55, 343, 379, 624, 651–654, 725]). The second one assures a one-to-one matching between feature pairs and discards correspondences if a feature in the first image is selected as best match for two or more features of the second image and vice versa (e.g. [5, 6, 529, 651, 652]).

Constrained matching enforces the epipolar constraint which states that for omnidirectional images each point observed in one image must lie on the epipolar curve in the other image (textbooks: [40, 131]).[5] Navigation strategies using this approach include [34, 54–56, 163, 164, 343, 529, 624, 652, 655, 725]. Only correspondences which can be explained by the epipolar constraint and the estimated motion parameters are kept, whereas other correspondences are discarded. As standard technique for detecting and removing outliers based on constrained matching, the *random sample and consensus* algorithm (RANSAC; Fischler and Bolles [178], reviews: [191, 563], textbook: [186, 278, 586]) is usually applied. For testing whether or not two places are identical, this means to discard all correspondences which cannot be explained by a pure rotation of the robot (i.e. a motion without translational component). Constrained-matching techniques are capable of accurately recognizing places, but are computationally too demanding to be executed for a larger number of image comparisons as it is required e.g. for localization (section 3.6.1.2). They are closely related to certain correspondence-based local visual homing methods relying on two-view stereo computations (section 3.5.2.3).

An alternative to these efficient matching methods are *staged matching methods*. Such methods apply a coarse-to-fine search combining signature-based and feature-based methods to recognize places. In the fist (coarse) step, a set of matching candidates is identified based on signature-based place recognition because signature-based techniques allow to efficiently perform a large number of image comparisons. For the small number of matching candidates remaining in the first step, computationally more demanding but also more accurate feature-based techniques are applied. Examples of staged matching techniques rely on statistical signatures [235–237, 476, 477] or on gist signatures [473, 475, 590]; on the second level, constrained matches are established.

Feature-based place representations are used for *feature-based visual compass methods* (section 3.4.2.3), for *correspondence-based homing methods* (section 3.5.2.3), and for mapping in combination with *sparse model-based maps* (section 3.6.3.1), *topo-metric maps* (section 3.6.3.2) and *purely topological maps* (section 3.6.4.1).

3.3.2. Role of Position Information

In the previous section, we introduced holistic, signature-based, and feature-based approaches to represent places based on omnidirectional visual information (sections 3.3.1.1 to 3.3.1.3). These approaches can be combined with different methods of using metrical position information; the resulting combinations are depicted in figure 3.16. The simplest case are *qualitative* navigation strategies (section 3.2.2) which do not incorporate metrical position information but solely use visual information for navigation. Qualitative navigation strategies can be implemented with any of the three methods for characterizing places based on visual information. *Quantitative* navigation

[5] The epipolar curve corresponds to the epipolar line for directed cameras. It is curved due to the distortions occurring for omnidirectional cameras.

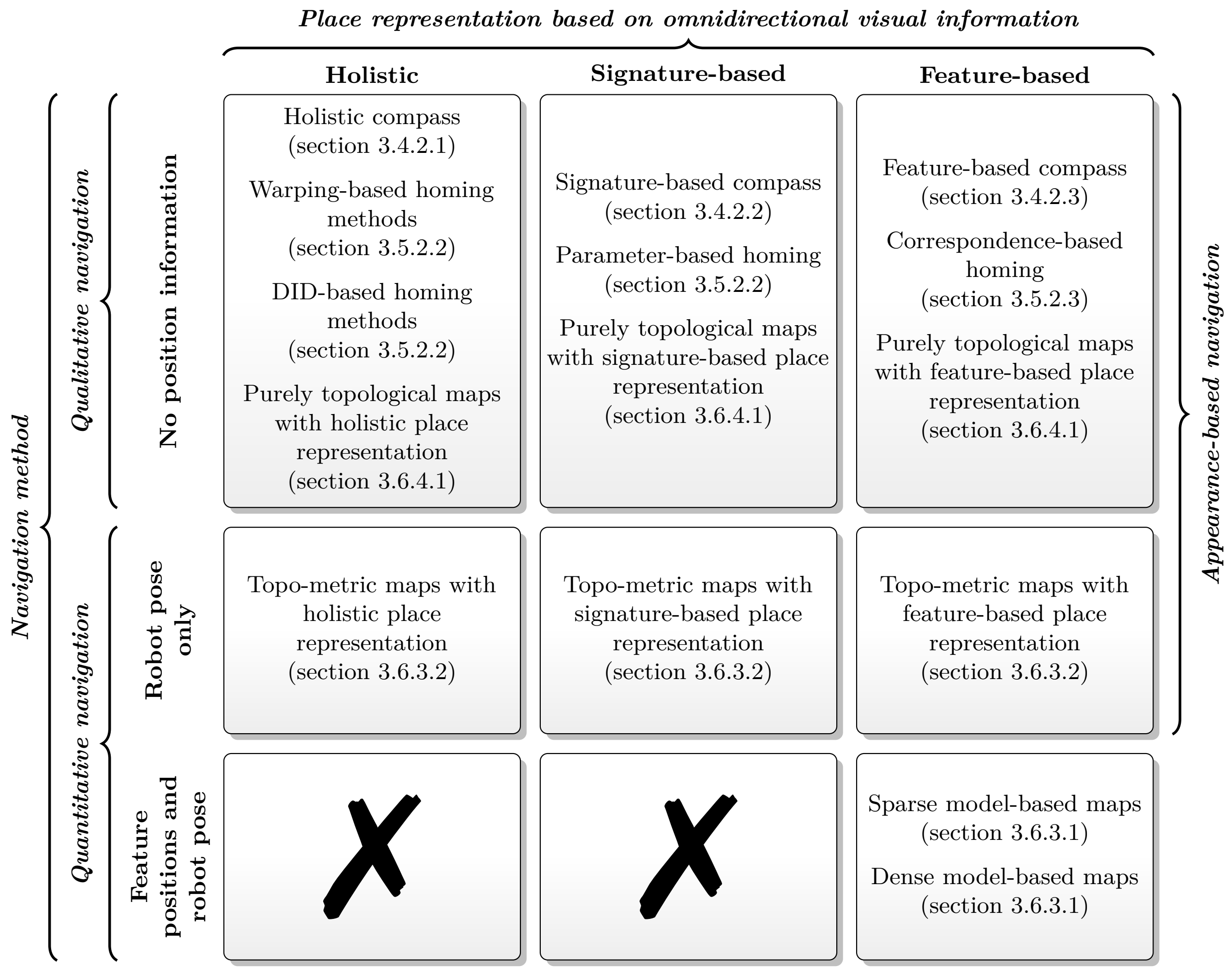

Place representation based on omnidirectional visual information
Holistic
Signature-based
Feature-based
Navigation method
Qualitative navigation
Quantitative navigation
No position information
Robot pose only
Feature positions and robot pose
Appearance-based navigation
Holistic compass (section 3.4.2.1)
Warping-based homing methods (section 3.5.2.2)
DID-based homing methods (section 3.5.2.2)
Purely topological maps with holistic place representation (section 3.6.4.1)
Signature-based compass (section 3.4.2.2)
Parameter-based homing (section 3.5.2.2)
Purely topological maps with signature-based place representation (section 3.6.4.1)
Feature-based compass (section 3.4.2.3)
Correspondence-based homing (section 3.5.2.3)
Purely topological maps with feature-based place representation (section 3.6.4.1)
Topo-metric maps with holistic place representation (section 3.6.3.2)
Topo-metric maps with signature-based place representation (section 3.6.3.2)
Topo-metric maps with feature-based place representation (section 3.6.3.2)
✗
✗
Sparse model-based maps (section 3.6.3.1)
Dense model-based maps (section 3.6.3.1)

Figure 3.16.: Combining visual place representations with metrical position information. As described in sections 3.3.1.1 to 3.3.1.3, visual information can be used to represent places by using the entire image (*holistic place representation*), by deriving a global image signature (*signature-based representation*), or by a set of features detected in the image (*feature-based representation*). Besides that, visual navigation methods can be categorized into *qualitative* and *quantitative* strategies. Whereas the former do not incorporate position information, the latter use position information and either estimate only the robot pose at the time of image acquisition or estimate the position of visible features and of the robot's pose. This leads to a theoretical number of nine different combinations, among which two combinations are impossible: for holistic and signature-based place representations position estimates cannot be assigned to visible features because these methods do not detect features but rather use the image as a whole. The corresponding methods are marked by crosses (✗). *Appearance-based methods* embrace navigation strategies which only rely on intensity information without assigning metrical position estimates to visible features. Thus, they include quantitative methods and qualitative methods which only estimate the robot's pose but not the spatial positions of visible features.

strategies involve position information and can be further categorized into (i) methods estimating *only the robot's pose* and (ii) methods estimating the *spatial positions of visible features and the robot's pose.* Methods estimating *only the robot's pose* at the time the visual information was acquired do not compute the spatial positions of visible features. Navigation strategies belonging to this class can rely on any of the three approaches to characterize places based on visual information. Navigation methods estimating the *spatial positions of visible features* are restricted to feature-based place representations. From the spatial positions of three or more visible features, the robot's pose can be derived based on the bearing towards these features (figure 3.17). Holistic or signature-based place representations cannot be used to estimate the spatial positions of visible features because

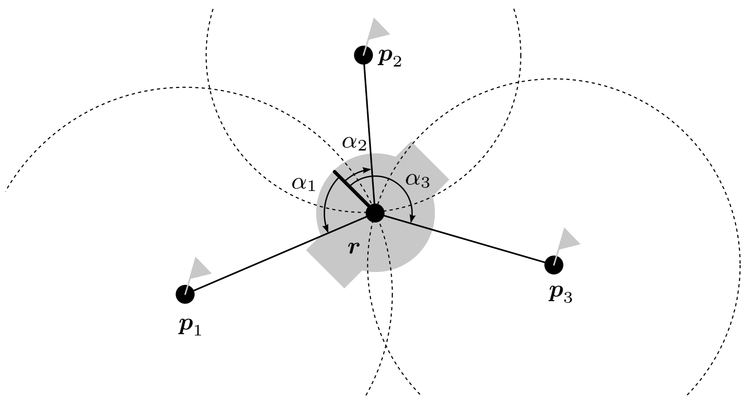


Figure 3.17.: Estimating the robot's pose (i.e. localization) with a bearing-only sensor. Based on the bearing angles α_1, α_2, and α_3 towards three or more visible features (or landmarks) $\boldsymbol{p}_1$, $\boldsymbol{p}_2$, and $\boldsymbol{p}_3$ with known positions (indicated by the flags), the pose (i.e. position and orientation) of the robot $\boldsymbol{r} = (x_r, y_r, \theta_r)^\top$ can be uniquely derived. The dashed circles around each feature $\boldsymbol{p}_i$ represent positions, for which the bearing towards the feature $\boldsymbol{l}_i$ corresponds to the angle α_i. The robot's true position $\boldsymbol{p}$ is at the unique intersection of the three circles.

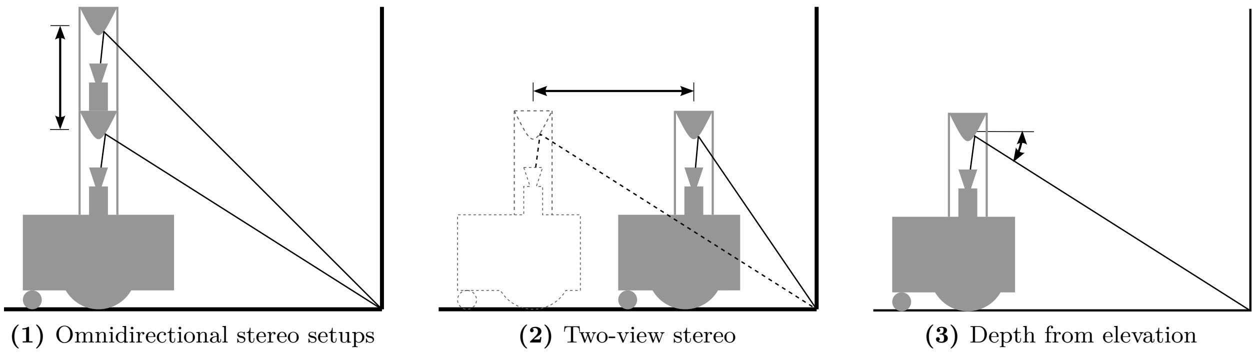


Figure 3.18.: Methods for determing range from omnidirectional images. Stereo methods (subfigures (1) and (2)) require two views and a distance estimate (referred to as baseline) to compute the range by triangulation. For omnidirectional stereo setups (subfigure (1)), the baseline is the vertical displacement of the cameras. For two-view stereo (subfigure (2)), the images are obtained from a monocular omnidirectional vision setup and the baseline is the distance between the positions of image acquisition. Only depth from elevation methods (subfigure (3)) allow for estimating the distance to an object from a single image. However, the range estimation is restricted to objects of constant height.

these representations use the entire image without feature detection. These two combinations are in (figure 3.16) marked by a cross.

3.3.2.1. Position Estimation Techniques

Position estimation requires to compute the distance of visible objects (not necessarily features) to the camera. As cameras are bearing-only sensors, the objects's distance to the camera cannot be estimated from only one camera image without further assumptions. Typically, two or more camera images are used to compute an objects's position in space. The images can either be taken with an omnidirectional stereo setup at the same position in space (figure 3.18.1) or with a monocular omnidirectional camera at two different positions in space (figure 3.18.2). Under the assumption that all objects have identical and constant height, the distance can be derived from a single monocular image. This technique is referred to as *depth from elevation* and exploits the vertical angle between the horizon and the visible object (figure 3.18.3).

To compute position estimates of visible objects or of the robot's pose and to maintain these estimates over time, most methods rely on position-estimation frameworks. Approaches to position estimation (figure 3.19) can be partitioned into *probabilistic methods* and *optimization-based methods*. Here, we only briefly recapitulate the different approaches of position estimation instead of giving a

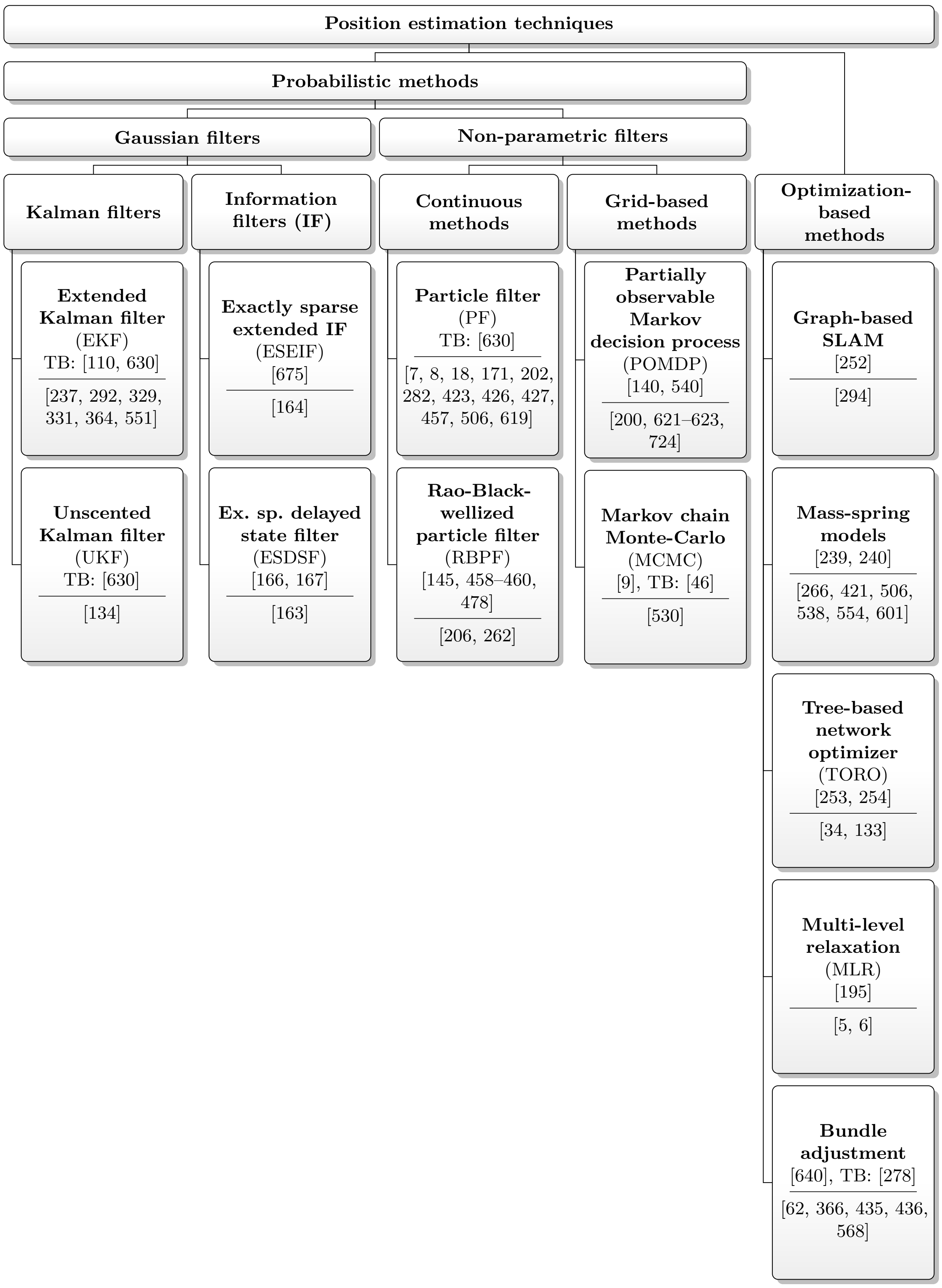

Position estimation techniques
Probabilistic methods
Gaussian filters
Non-parametric filters
Kalman filters
Information filters (IF)
Continuous methods
Grid-based methods
Optimization-based methods
Extended Kalman filter (EKF) TB: [110, 630]
[237, 292, 329, 331, 364, 551]
Unscented Kalman filter (UKF) TB: [630]
[134]
Exactly sparse extended IF (ESEIF) [675]
[164]
Ex. sp. delayed state filter (ESDSF) [166, 167]
[163]
Particle filter (PF) TB: [630]
[7, 8, 18, 171, 202, 282, 423, 426, 427, 457, 506, 619]
Rao-Blackwellized particle filter (RBPF) [145, 458–460, 478]
[206, 262]
Partially observable Markov decision process (POMDP) [140, 540]
[200, 621–623, 724]
Markov chain Monte-Carlo (MCMC) [9], TB: [46]
[530]
Graph-based SLAM [252]
[294]
Mass-spring models [239, 240]
[266, 421, 506, 538, 554, 601]
Tree-based network optimizer (TORO) [253, 254]
[34, 133]
Multi-level relaxation (MLR) [195]
[5, 6]
Bundle adjustment [640], TB: [278]
[62, 366, 435, 436, 568]

Figure 3.19.: Overview of position estimation techniques used with omnidirectional visual information for navigation of wheeled robots. For exhaustive lists of such techniques, please refer to the textbooks [46, 630] and the reviews [25, 140, 150, 151, 187, 631, 676]. For each estimation technique, two types of references are given: the references above the horizontal line refer to textbooks (TB) or the original papers. References below the horizontal line are examples of omnidirectional visual navigation methods applying the corresponding technique. Please note that the given lists of examples are not exhaustive.

detailed description; for these details, the reader is referred to the textbooks [110, 630], to reviews [25, 150, 151, 194, 631], or to the papers given in figure 3.19.

Probabilistic Techniques

Methods of this class are in most cases Bayesian filtering techniques following a predictor-corrector scheme (theoretical reviews: [140, 151, 188, 676], application to robot navigation: [99, 150, 151, 631], robotics textbooks: [110, 187, 586, 630]). The robot's current state is predicted based on internal sensor data (in most cases odometry information) and corrected by visual sensor information as an external sensor cue. The prediction step increases the estimation uncertainty, whereas the correction step improves the estimate and reduces uncertainty. Depending on how probabilistic methods represent the belief about the robot's state, *Gaussian filters* and *non-parametric filters* can be distinguished. *Gaussian filters* represent the belief about the robot's state by Gaussian probability distributions. The various subtypes of Kalman and information filters all belong to this class. *Non-parametric filters* lift the restriction of representing the robot's belief by a uni-modal Gaussian distribution and thus can deal with arbitrary probability distributions. In most cases, the belief distribution is approximated by a set of particles drawn randomly from the underlying probability distribution. Examples include particle filters and derivatives thereof for the continuous case and Markov decision processes for the discrete or grid-based case.

Since the described filtering techniques incrementally estimate the robot's state in every time step, difficulties arise for methods also estimating the position of visible features because —at least with a standard omnidirectional vision setup providing only bearing information— the features' distance from the camera cannot be estimated with only one camera image. To circumvent this drawback, most methods use *delayed* updates (Lemaire and Lacroix [364], reviews: [99, 341]). They update the map only if the feature's position can be reliably estimated by triangulation, i.e. after some time of tracking the feature. Nevertheless, methods relying on *undelayed* measurements exist, but they (i) require a special treatment of the immediate update if used with bearing-only sensors (Lemaire and Lacroix [364], directed vision: e.g. [309, 310, 360, 361]), or (ii) have to rely on range information obtained from an omnidirectional stereo sensor (e.g. [119, 331, 332, 443, 485]).

Optimization-based Methods[6]

Optimization-based approaches to position estimation search for the optimal map configuration by a constrained optimization process ([25, 150, 194, 631], tutorial: [252], textbooks: [110, 630]). The robot's former and current positions and —in case of model-based maps— also the position of sensed features are constrained by the sensor measurements gathered over time. This optimization process is usually referred to as *relaxation* (textbook: [630], review: [194, 252, 432]). As the resulting system of equations is usually sparse, special and more efficient optimization algorithms can be applied (review: [631]). The term "graph-based" resulted from visualizing the system of equations: robot (and feature) positions can be represented as graph nodes and sensor measurements can be understood as links. Optimization-based approaches scale better to large-scale environments than probabilistic methods based on Kalman filters (reviews: [507, 631]). Like probabilistic methods, visual graph-based methods require at least two observations of a certain feature (or two constraints) to estimate its position. Although early works in this field were offline solutions [630, 631] first gathering all observations and afterwards computing the map (thus making all updates delayed), recent improvements can achieve graph optimization in real-time allowing for optimizations whenever new features are added to the map or whenever loops are closed [252]. Examples of optimization frameworks and pointers to corresponding literature are given in figure 3.19.

[6]Lorenz Hillen is grateful to Jochen Sprickerhof (Knowledge-Based Systems Research Group, Institute for Computer Science, Osnabrück) for fruitful and insightful discussion on this section.

3.3.3. Discussion of Place Representation and Recognition

In this section, we discuss place representation and recognition based on visual information. The considered aspects include the choice of a particular representation (section 3.3.3.1), its role to achieve robustness against perceptual variability (section 3.3.3.2), its relation to biological and psychological research on visual navigation (section 3.3.3.3), and the application of place representations for this work (section 3.3.3.4).

3.3.3.1. Choosing a Place Representation

The representation of places is an essential building block for all higher-level navigation capabilities ranging from visual compass methods to mapping (figure 3.2). The choice of a particular representation strongly depends on the robot's application or task and also influences many aspects of more complex navigation capabilities. When choosing a place representation, the first decision is whether or not the robot's task requires position information. In case quantitative navigation is required or reasonable, one has to further decide if position information for visible features is required or if it sufficient to only estimate the robot's pose. The former case allows to reconstruct or visualize the robot's environment (structure from motion or scene reconstruction; section 3.6.1.2) but requires a feature-based representation of places. The latter case only allows to visualize the robot's trajectory, but it can be more flexible because it can be used with holistic, signature-based, or feature-based representations of places. In general, position information can disambiguate different places with identical or similar visual appearance thus reducing the likelihood of perceptual aliasing (section 3.2.3.1). If position information is not necessary to accomplish the robot's task, qualitative navigation methods can be applied. Qualitative methods can be used with all place representations and are often computationally more efficient than quantitative methods.

The second question is to decide for the best suited representation of places among the three possibilities, namely *holistic*, *signature-based*, and *feature-based* representations. This decision strongly depends on the available computing power and storage capacity. Signature-based approaches are parsimonious w.r.t. computing power and memory requirements. Often, signatures with a dimensionality comparable to the dimensionality of a single feature descriptor used for feature-based representations is used (SIFT and SURF descriptors are both 128-dimensional; Bay, Tuytelaars, and Van Gool [33] and Lowe [389]). The computational and memory efforts of holistic methods strongly depend on the size of the images used to represent a place. With a typical resolution of 1° per pixel, these methods require moderate memory and computing power. The memory requirements of feature-based approaches strongly depend on the number and on the dimensionality of the feature descriptors used to represent an image. Regarding computing time, feature-based methods currently applied for navigation based on omnidirectional vision are computationally more demanding than the other possibilities to represent places. However, beginning with SURF (Bay, Tuytelaars, and Van Gool [33]) developed as a faster alternative to SIFT features (Lowe [388, 389]), a series of fast feature detectors and descriptors has been proposed: features from accelerated segment test (FAST; Rosten and Drummond [542] and Rosten, Porter, and Drummond [543]), binary robust independent elementary features (BRIEF; Calonder et al. [77]), oriented fast and rotated BRIEF (ORB; Rublee et al. [547]), binary robust invariant scalable keypoints (BRISK; Leutenegger, Chli, and Siegwart [365]), and fast retina keypoints (FREAK; Alahi, Ortiz, and Vandergheynst [2]). These descriptors are binary strings which are computed from a set of comparisons between pixel intensities within a small image region and which can be very efficiently compared by the Hamming distance (e.g. [469]). Thus, the descriptors can be both computed and compared much faster than traditional vector-based descriptors. To the best of our knowledge, these new feature descriptors have not yet been applied for robot navigation based on omnidirectional vision, but we expect their application to be only a matter of time.

3.3.3.2. Robustness Against Perceptual Variability

Since the representation of places based on visual information is a building block for more complex navigation strategies, it plays an essential role for achieving robustness against illumination changes and dynamic changes of the scene (section 3.2.3.2). For holistic and signature-based place representations, robustness against illumination changes can be achieved by image preprocessing or illumination-tolerant dissimilarity functions; in case of signature-based approaches, illumination invariance —or at least tolerance— can additionally be achieved by choosing an appropriate signature function for computing the low-dimensional global image descriptor. For feature-based approaches, illumination invariance is a property of the used feature descriptors. Thus, independent of the illumination conditions, feature descriptors can be compared with simple standard dissimilarity functions (reviews: [19, 91, 92, 225, 635]) which do not tolerate illumination changes.

Robustness against dynamic scene changes is more difficult to achieve, especially if the scene changes strongly influence the visual appearance of an image. Methods operating on holistic or signature-based place representations can compensate smaller changes of the visual appearance, but it is likely that strong changes of the appearance cannot be compensated and cause the navigation strategy to fail. For feature-based methods, changes of the visual appearance can result in erroneous or ambiguous matches. As long as a sufficient number of correct matches remains after eliminating outliers or ambiguous matches, feature-based methods can cope with dynamic scene changes. If the number of remaining matches becomes too small, navigation is likely to fail.

3.3.3.3. Place Representations and Spatial Cognition

Navigation is an essential behavior for both robots and living beings, necessary for surviving in their "everyday life" and for accomplishing more complex tasks (e.g. [403, 596, 597, 677, 678]). Here and in sections 3.4.3.2, 3.5.3.1 and 3.6.5.2, we will briefly recapitulate the current state of related research in the field of spatial cognition, which embraces ethological research on animal navigation and psychological research on human navigation capabilities. We discuss these issues because (i) some methods and applications mentioned in sections 3.6.3 and 3.6.4 are biological models or bio-inspired navigation strategies, (ii) the methods proposed in chapters 4 to 6 of this thesis rely on bio-inspired building blocks for achieving robust navigation of an autonomous cleaning robot, and (iii) because of their large field of view and the relatively small resolution omnidirectional vision sensors are often compared with the complex eyes of insects (section 3.2.4.1).

The snapshot hypothesis of local visual homing (section 3.5.1 and figure 3.22) states that insects use the entire visual panorama for navigation. 30 years after the hypothesis was suggested by Cartwright and Collett [87] and Wehner and Räber [680], evidence exists that insects indeed represent places by the entire visual panorama (e.g. [247, 248, 702, 703, 717]). Ongoing research on insect visual homing investigates, how visual information is processed and which information of the entire panorama is used. Currently, two main hypotheses are discussed: (i) the skyline panorama resulting from the segmentation between sky and ground (e.g. [31, 247, 248]) and (ii) dynamic snapshots based on optical flow patterns [141, 142]. Details of these hypotheses are discussed in section 3.5.3.1. Beyond the hypothesis that insects use the entire panorama for navigation, it is at the current state of research not possible to further conclude whether insects rely on a representation comparable to holistic or to signature-based approaches. Although insects are able to react to salient patterns such as color or texture (reviews: [20, 232]), which could be interpreted to be closely related to features used in computer vision, it is unlikely that such patterns are used by insects to represent places in a way similar to the feature-based representations used in robotics (reviews: [702, 703]).

Early studies on navigation of mammals often concluded that depth information is the essential cue to represent places (e.g. [100], reviews: [101, 702, 703]). Only recently, results are explained by matching entire images (rats: [103, 272, 273, 605], humans: [227]). To this end, it is likely that mammals also use the entire snapshot to characterize places. However, due to the plethora of cues and strategies used by rats and humans for navigation, other representations cannot be definitely ruled out. Information processing steps in the human and mammal visual system include to detect salient regions differing from the surrounding. Computational models mimicking these processing steps are referred to as *visual saliency* or *visual attention* mechanisms (review: [304]). Theoretically, such salient points could play a similar role as local image features play for computer vision. Saliency points are relevant for object detection in mammals, but their role for navigation is unclear [703].

3.3.3.4. Application of Place Representations in This Work

Throughout this thesis we rely on a holistic representation of places. We opted for this representation because it allows to apply a 2D warping method for local visual homing, which is an accurate and efficient homing method developed by our group (sections 3.5.2.2 and 4.4.2). Furthermore, it allows to easily integrate images representing already cleaned areas into a dense topo-metric map of the robot's environment (sections 3.6.3.2 and 4.2 and chapters 5 and 6). Since the visual detection of already cleaned areas (loop-closure detection) involves comparing the currently perceived image with a potentially large number of images stored in the map, we suggest an additional signature-based representation (chapter 6). This approach allows for more efficient image comparisons saving computing time for other tasks required to autonomously clean complex-shaped areas. Although feature-based approaches have to be considered the current standard methods in robot vision and although large achievements were made with feature-based methods during the period of time considered in this review chapter, we do not consider such navigation strategies. We are of the opinion that the computational power of an autonomous floor-cleaning robot for domestic usage is not sufficient for real-time operation on larger dense topo-metric maps if a feature-based representation of places is used.

3.4. Visual Compass

Together with local visual homing strategies, visual compass methods form the second level of the proposed navigation hierarchy (section 3.2.2 and figure 3.2). This level is also referred to as *local navigation* because it derives information from two images acquired at nearby positions in space. In the following, we define (section 3.4.1), review (section 3.4.2), and discuss (section 3.4.3) visual compass methods.

3.4.1. Definition of Visual Compass

Visual compass methods estimate the change of the robot's orientation between two images acquired at identical or nearby positions in space (figure 3.20). In the following, we will stick to the notation of local visual homing (section 3.5.1) and refer to one image as *current view* and to the other image as *snapshot*. Visual compass methods can be understood as partial ego-motion estimation techniques which are only capable of recovering the rotational motion component between current view and snapshot but not the direction of translation and its absolute length. The estimate is also referred to as as *compass shift*. This term is influenced from wheeled robots operating in the plane and relying on panoramic images: in this case, a change of the robot's orientation results in a horizontal shift of the image (figure 3.20), but the visible image content does not change (section 3.2.4.2). If snapshot and current view are acquired at different positions in space, there is not only a rotational but also a translational motion between the robot's poses. Hence, visible objects appear in the image not only shifted but also scaled. Up to a certain degree, visual compasses can tolerate such

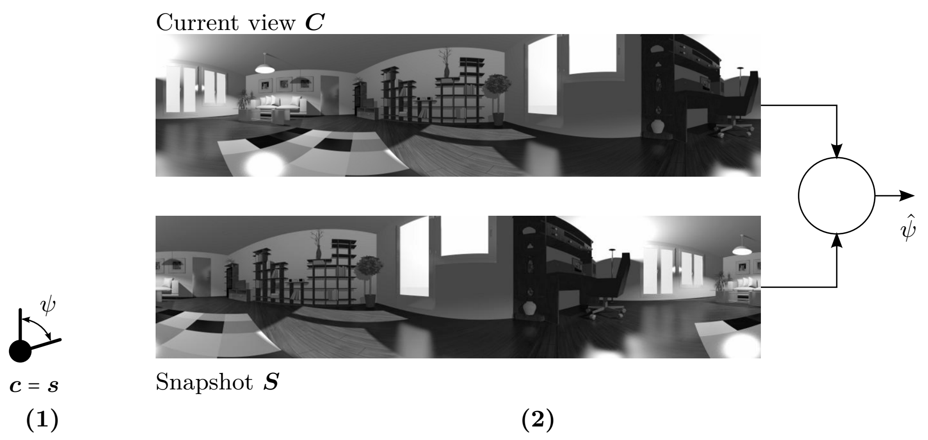

Current view $\boldsymbol{C}$
$\hat{\psi}$
ψ
$\boldsymbol{c} = \boldsymbol{s}$
Snapshot $\boldsymbol{S}$
(1)
(2)

Figure 3.20.: Principles of visual compasses. Subfigure (1): situation in space; subfigure (2): sketch of the algorithm. In the depicted case, two images $\boldsymbol{C}$ and $\boldsymbol{S}$ were acquired at identical positions ($\boldsymbol{c}$, $\boldsymbol{s}$) but with different orientation (solid lines). The compass method (circle) compares two images in order to derive an estimate $\hat{\psi}$ of the robot's true change of orientation ψ. The depicted case is a holistic compass method comparing images as a whole, but signature-based and feature-based compasses follow the same principle (section 3.4.2).

translational image changes, but the accuracy of the estimates decreases with increasing spatial distance between the snapshots. If the robot's task involves not only to estimate the rotational but also the translational component between two robot poses, a compass method in combination with local visual homing or a local visual homing method estimating both motion components has to be applied (section 3.5.1).

Estimates of the compass shift between two images are needed for the following three applications. First, to physically align the robot w.r.t. a reference direction in order to eliminate the orientation difference between the images. By this means, the robot can e.g. be guided along a known route (sections 3.6.3.2, 3.6.4.1 and 3.6.4.2). The current view corresponds to the currently perceived image, and the snapshot is an image acquired at an intermediate goal along the route. Second, to align images w.r.t. a common reference direction without physically moving the robot. In this sense, compass methods are applied prior to local visual homing if the used homing method cannot operate on arbitrarily aligned input images (section 3.5.2). Images are then aligned by horizontally shifting one of the omnidirectional images such that the compass shift is compensated. This *mental rotation* of one of the images corresponds to a rotation of a mobile robot (section 3.2.4.2). The third application includes to estimate spatial relations in a topological or topo-metric maps (sections 3.6.3.2 and 3.6.4.1). For applying compass methods in this sense, current view and snapshot are two images acquired at former robot positions which do in most cases not correspond to the image perceived at the robot's current position.

3.4.2. Literature Review on Visual Compass Methods

Depending on how visual information is used to characterize places (section 3.3), compass methods can be categorized into *holistic methods*, *signature-based methods*, and *feature-based methods* (sections 3.4.2.1 to 3.4.2.3 and figure 3.21). The only prerequisite is that the used representation of places has to be dependent on the robot's orientation. This aspect has to be considered for signature-based and feature-based representations.

3.4.2.1. Holistic Compass Methods

Holistic compass methods operate on holistic representations of places (section 3.3.1.1) and use the entire intensity image or a preprocessed variant of it to derive the orientation estimate. All

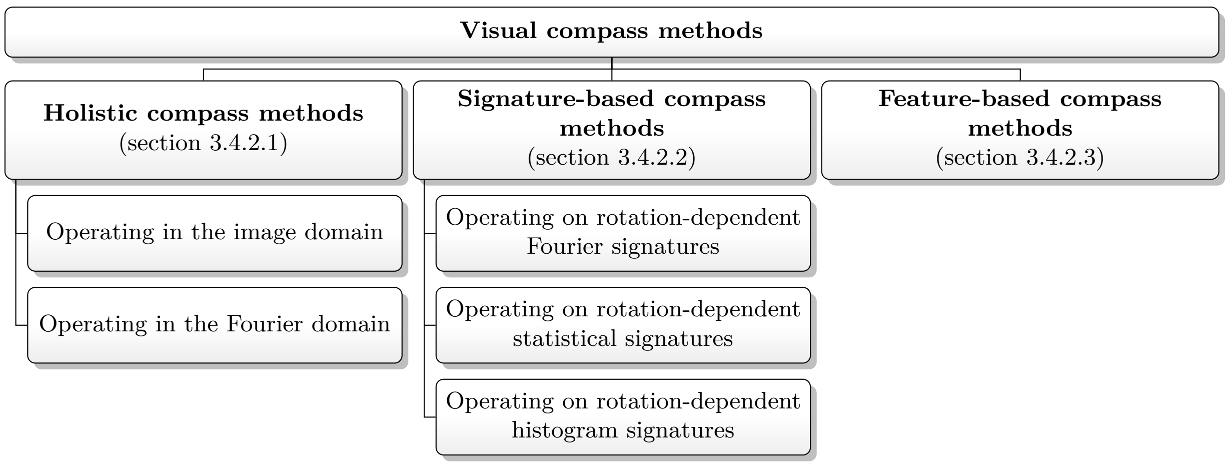

Visual compass methods
Holistic compass methods (section 3.4.2.1)
Signature-based compass methods (section 3.4.2.2)
Feature-based compass methods (section 3.4.2.3)
Operating in the image domain
Operating in the Fourier domain
Operating on rotation-dependent Fourier signatures
Operating on rotation-dependent statistical signatures
Operating on rotation-dependent histogram signatures

Figure 3.21.: Categorization of visual compass methods.

methods of this class are influenced by the seminal paper Zeil, Hoffmann, and Chahl [718] which derives the estimate from step-by-step shifting one of the images, comparing the images by applying an image dissimilarity function (reviews: [19, 91, 92, 225, 635]), and searching for the best match. Holistic compass methods can be categorized into methods operating in the *image domain* and methods operating in the *Fourier domain* depending on how they compare images in order to determine the compass shift.

Compass methods operating in the *image domain* include the original compass by Zeil, Hoffmann, and Chahl [718] and improvements thereof. These include improvements of its robustness against illumination changes [612] and of its compass accuracy [353, 354, 604, 662]. The method by Labrosse [353, 354] only considers regions in the panoramic image which allow for reliably estimating the change of orientation, i.e. the rotational motion component between the considered images. These are the image regions viewing into the robot's forward and backward directions; regions viewing into lateral directions do not allow to disambiguate translational and rotational motion components (e.g. [129, 486]). The methods [604, 662] gather information from several images in order to exclude unreliable image regions from the estimation process. The algorithm by [604] excludes image regions containing nearby objects, and the method by [662] discards nearby and homogeneous image regions. Both methods exploit the fact that orientation is best estimated from distant objects.

The second group of holistic compass methods exploits that image correlations can be efficiently computed in the Fourier domain (textbook: [65]) and that the compass shift is identical to the shift of the maximum correlation value. Compass methods following this principle were proposed for 1D panoramic images [75, 610], 2D panoramic images estimating rotations in two directions [75], and hemispherical images (figure 3.5.2) estimating 3D rotations [296, 399–401, 571, 572]. In the latter case, spherical harmonics, i.e. Fourier transformation with spherical basis functions [146, 398], were applied. The accuracy of the compass depends on the number of used Fourier coefficients. A small number results in a coarse estimate, and a larger number of coefficients yields more accurate estimates [296, 400, 571, 572, 609]. Using only a small number of Fourier coefficients makes the method closely related to signature-based techniques (section 3.4.2.2). The holistic visual compass method proposed in section 5.2.3 also belongs to this class of compass methods.

3.4.2.2. Signature-Based Compass Methods

Signature-based compass methods derive the estimate of the robot's orientation change from comparing two low-dimensional signatures (section 3.3.1.2) derived from the corresponding omnidirectional images. The low dimensionality of the signatures not only allows for efficient image comparisons but also for implementing efficient compass methods. However, the change of the robot's orientation can only be estimated from *rotation-dependent* signatures (section 3.3.1.2). Because most of the navigation strategies relying on signatures to characterize places apply rotation-invariant signatures, signature-based compasses only play a minor role.

For Fourier signatures including phase information, the orientation change can be estimated from the phase difference. This technique is applied by [172] and is closely related to the holistic compass methods operating in the Fourier domain (section 3.4.2.1). Compass methods are proposed for rotation-dependent statistical signatures [256, 257, 340] and for rotation-dependent histogram-based signatures [261]. For both types of signatures, each entry of the signature corresponds to a certain vertical slice of the omnidirectional image (figure 3.13.2). This allows to apply principle of the compass method by Zeil, Hoffmann, and Chahl [718]: one of the signatures is shifted step-by-step and repeatedly compared to the other signature, and the compass shift is determined by searching for the shift leading to the best match.

3.4.2.3. Feature-Based Compass Methods

In case images are represented by a set of visible features (section 3.3.1.3), the compass estimate can be derived from correspondences established between the image features of both images. Feature-based compass methods need to incorporate the image positions of visible features because only the feature position in the image depends on the robot's orientation. Matching methods such as *bag of features* or *bag of words* methods (section 3.3.1.1) only taking common features but not their image position into account are rotationally invariant and are therefore not suited for feature-based compasses.

The compass methods by [5, 6, 560] simply derive the estimate from the horizontal displacement of the matched features. The compass algorithms proposed by [408–412] are more elaborated but also computationally more complex. The methods rely on stereo relations between images and can derive the estimate directly from matching parallel lines [410, 412] and by step-by-step shifting one of the images until the shift is compensated. In principle, the correspondences could also be used to estimate the essential matrix and to determine the rotational component from matrix factorization. However, factorizing the essential matrix not only yields the rotational motion component but also the translational component, and is therefore considered to be a correspondence-based *homing* method (section 3.5.2.3).

3.4.3. Discussion of Visual Compass Methods

In this section, we discuss the relevance of visual compass methods (section 3.4.3.1), their relation to spatial cognition (section 3.4.3.2), and their application in this work (section 3.4.3.3).

3.4.3.1. Relevance of Visual Compass Methods

The relevance of visual compasses for navigation of wheeled mobile robots relying on omnidirectional images is often neglected. This is probably due to a classical perspective of navigation methods which estimate the spatial positions of visible features (sections 3.3.2.1 and 3.6.3.1). As such methods allow for deriving the robot's pose from visible features (figure 3.17), the orientation change between can be computed directly from the *pose estimates*. Thus, compass methods are not needed with such methods. Only with the advent of appearance-based methods, which do not estimate spatial positions of visible features (section 3.2.2), deriving compass information directly from the *visual image information* became relevant. Recently, route-following methods relying only on compass information were proposed (sections 3.6.3.2, 3.6.4.1 and 3.6.4.2). Due to their increased relevance, we opted to discuss compass methods separate from local visual homing. Since the original application of compass methods was to align images prior to homing computations, earlier reviews such as

Möller, Krzykawski, and Gerstmayr [451] or Zeil [717] cover compass methods together with local visual homing methods.

3.4.3.2. Visual Compass Methods and Spatial Cognition

Compass information is an essential cue for animal navigation. Common senses include magnetic compasses (birds: [45, 420, 466, 692, 693, 716], monarch butterflies: [534], sea turtles: [383]), or polarization compasses (insects: [288], birds: [467, 468, 693, 694], monarch butterflies: [534]). The only evidence that animals can rely on a compass similar to holistic compass methods for robot navigation are the works by [637, 638]. The authors conclude that bees derive orientation information from a static snapshot as a backup compass if their solar compass fails because of overcast weather conditions. Regarding mammals or humans, we are not aware of experiments which could show that visual compass information comparable to the compass methods reviewed in this section is used for navigation.

3.4.3.3. Application of Visual Compass Methods in This Work

In the context of this thesis, visual compass methods are applied for visually detecting already cleaned areas (i.e. for loop-closure detection) with methods operating on a holistic representation of places (chapter 5). We rely on the compass method proposed by Zeil, Hoffmann, and Chahl [718] and systematically evaluate combinations of image preprocessing and image comparison techniques (i) to increase the robustness against illumination changes and (ii) to reliably detect already cleaned areas (or loop closures, respectively). In addition, we propose a new holistic compass method operating in the Fourier domain, which follows the same principles as the methods by [75, 610]. For signature-based loop-closure detection (chapter 6), we rely on rotation-invariant signatures which do not require a visual compass. As the min-warping method for local visual homing (sections 3.5.2.2 and 4.4.2) applied in this thesis can operate with arbitrarily aligned images, we do not need to apply visual compass methods prior to home-vector computations.

3.5. Local Visual Homing

In this section, we introduce local visual homing as technique (i) for navigating from the robot's current position back to a closeby goal position and (ii) for taking the bearing and (in some cases) for estimating the compass change between the considered images without physically moving from one place to the other. The section is structured similar to sections 3.3 and 3.4: section 3.5.1 defines local visual homing, section 3.5.2 gives a literature on related work of this field, and it ends with a discussion (section 3.5.3).

3.5.1. Definition of Local Visual Homing[7]

Local visual homing, which is in some cases also referred to as guidance [189, 642], is the ability to return to a previously visited place based solely on visual information. Visual homing methods are strongly influenced by the snapshot hypothesis of insect navigation (Cartwright and Collett [87] and Wehner and Räber [680]). It states that places are characterized by images taken at their corresponding positions. By comparing its currently perceived image, usually referred to as *current view*, and the image stored at the home position, referred to as *snapshot*, a movement direction is computed, which guides the agent into the direction of its goal (figure 3.22). Mathematically, this movement direction can be described by a vector, the *home vector*, pointing from the current position to the snapshot position. Local visual homing methods are only capable of computing

[7]This section is an extension of section 4.1 in our journal publication Gerstmayr-Hillen et al. [222]

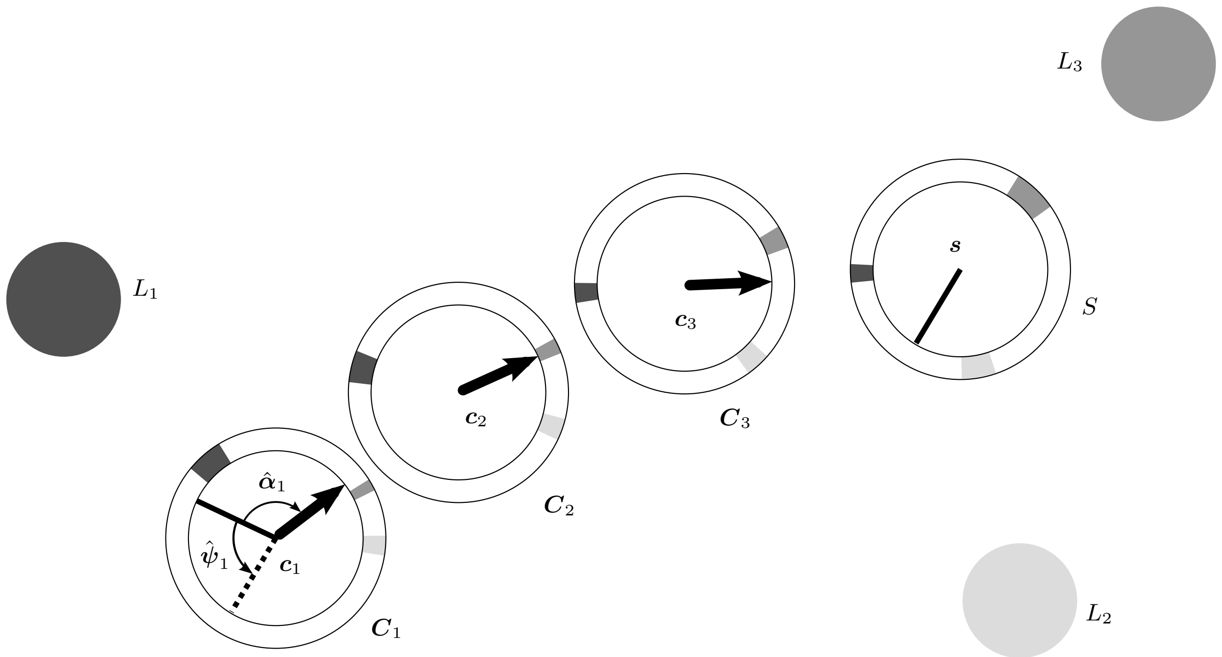


Figure 3.22.: Principles of local visual homing. The robot is initially located at position $\boldsymbol{s}$ in a certain orientation (solid black line). It memorizes the currently perceived image $\boldsymbol{S}$ as snapshot. Panoramic images are depicted as one-dimensional images showing the three surrounding objects or landmarks L_1, L_2, and L_3. After exploration, the homing process starts at position $\boldsymbol{c}_1$. By comparing the current view $\boldsymbol{C}_1$ and the snapshot $\boldsymbol{S}$, local visual homing computes the two estimates $\hat{\psi}_1$ and $\hat{\alpha}_1$. The former is an estimate of the change between the robot's current (solid black line) and former (dashed) orientation; the latter is an estimate of the home direction. In terms of ego-motion estimation, these two estimates correspond to the rotational component and the translational component up to scale between the two images. By following the home vector (black arrow), the robot is guided towards the snapshot position $\boldsymbol{s}$. The influence of erroneous estimates can be reduced by repeating this procedure (in the depicted case at positions $\boldsymbol{c}_2$ and $\boldsymbol{c}_3$). While approaching the snapshot position $\boldsymbol{s}$, the current views $\boldsymbol{C}_i, i \in \{1,2,3\}$ become more and more similar to the snapshot $\boldsymbol{S}$. In the context of this thesis, local visual homing is not used for guiding the robot towards a goal position, but for estimating angular relations between two positions in space.

the direction of the vector but not its length. Thus, they can be considered as a partial solution of the ego-motion estimation problem (textbook: [641], review: [130]). Algorithms for solving the ego-motion estimation problem recover the translational and the rotational component of the robot's movement between two positions in space. The direction of the translational component corresponds to the home vector direction but only up to scale; the rotational component corresponds to the compass shift, i.e. the change of the robot's orientation between current view and snapshot (section 3.4) prior to homing. Only some homing methods are capable of estimating both the home direction and the compass shift. Others assume current view and snapshot to be aligned w.r.t. a common reference direction; this can be accomplished by applying a compass method (section 3.4).

To return to the snapshot position, the robot follows the direction of the home vector and —to compensate for erroneous estimates— repeats the homing step until it reaches its goal position (figure 3.22). While approaching the goal position, the intermediate images $\boldsymbol{C}_1$, $\boldsymbol{C}_2$, and $\boldsymbol{C}_3$ perceived at positions $\boldsymbol{c}_1$, $\boldsymbol{c}_2$, and $\boldsymbol{c}_3$ become more and more similar to the snapshot $\boldsymbol{S}$. By this means, the robot brings the currently perceived view into accordance with the snapshot and is guided back to its goal position. In the homing process, places are solely characterized by the image of the surrounding objects or landmarks (L_1, L_2, and L_3). Thus, local visual homing does not involve metrical position information (hence it is a qualitative navigation method; section 3.3.2).

Due to operating on two positions in close vicinity to each other, local visual homing methods belong to the class of *local navigation methods* (section 3.2.2). Their navigation range is restricted to the robot's sensory horizon (Franz and Mallot [189]), and every snapshot position is surrounded by a *catchment area*. For positions within this area, homing to the snapshot position is possible; for positions outside, homing will fail. The navigation range can be extended by integrating several snapshots into a map representing the robot's workspace (section 3.6).

In the context of this thesis, homing is not used for actively guiding the robot towards a stored snapshot position. Rather, visual homing is used to take the bearing from the current robot position to a former robot position represented by the corresponding snapshot stored in the topological map. By this means, former robot positions are used as landmarks, and spatial relations between different landmarks can be derived. As local visual homing does not involve position information, these angular relations only define the spatial arrangement of the considered snapshots up to scale. The absolute scale can be recovered by introducing distance information, for example derived from the robot's odometry (sections 3.6.3.2 and 4.3.3.3).

3.5.1.1. Relation to Visual Servoing

So far, we introduced local visual homing from a rather behavior-oriented perspective. More control-theoretical solutions to local visual homing have evolved in the broad field of *visual servoing* embracing navigation strategies for moving a robot from a current configuration to a desired configuration. The feedback required for controlling the robot is derived by means of computer vision (reviews: [95–97, 299]). All methods have in common that they are strongly influenced by control theory (textbook: [143], tutorials: [303, 650]). Visual servoing can be categorized into *position-based* and *image-based* visual servoing. *Position-based* visual servoing includes (i) to reconstruct a geometrical model of the robot's environment, (ii) to estimate the current camera pose, and (iii) to thereupon derive control commands to transform the current robot configuration into the desired configuration. Due to those processing steps, position-based visual servoing is closely related to mapping methods reconstructing the robot's environment (model-based maps, section 3.6.3.1). As this involves metrical position information, position-based approaches to visual servoing are quantitative navigation strategies (section 3.2.2). In contrast, *image-based visual servoing* does not reconstruct the robot's environment but rather operates directly on the images. As these methods do not incorporate metrical position information, they are qualitative navigation strategies (section 3.2.2). Algorithms for image-based visual servoing establish correspondences between visible features detected in the current view and snapshot. Therefore, they rely on a feature-based representation of places (section 3.3.1.3). Control commands are derived in order to reduce differences between the current and the desired configuration by minimizing the displacement of matching feature pairs. Visual servoing methods (both image-based and position-based) exist for control of robot manipulators and for navigation of wheeled mobile robots. For the latter, algorithms relying on standard cameras (see the reviews [95–97, 299] for examples) and on omnidirectional vision (e.g. [265, 408, 409, 411, 434]) were proposed.

Local visual homing and visual servoing methods solve the same task, namely guiding a robot from one configuration to a desired configuration. In the context of navigation of wheeled robots moving in the plane, configuration and pose can be used interchangeably. Local visual homing is traditionally used for local navigation of mobile robots relying on omnidirectional vision as primary sensory information. The focus is on deriving estimates of the home direction and the compass shift. Although, it is strongly influenced by the snapshot hypothesis and by modeling animal behavior (section 3.5.3.1), visual homing methods relying on holistic, signature-based, and feature-based representations of places exist. In contrast, visual servoing methods are rooted in control theory, solely rely on a feature-based representation of places, and are applicable to both robot manipulators and mobile robots with different kinds of sensors. Thus, visual servoing is a much wider field than local visual homing. The focus of many visual servoing methods is not on estimating home direction and compass shift, but on finding suitable control laws to reduce the deviation between current and desired pose. Nevertheless, the particular subdomain of image-based visual servoing for wheeled

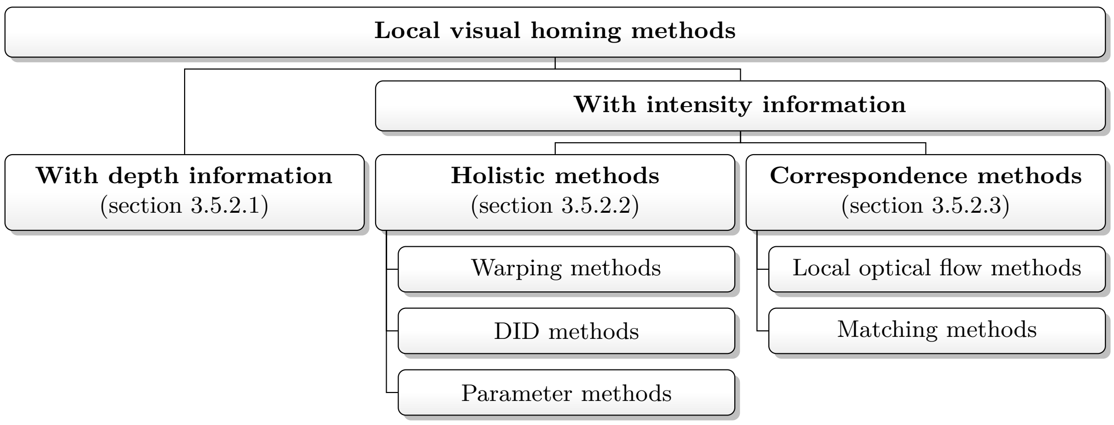

Local visual homing methods
With intensity information
With depth information (section 3.5.2.1)
Holistic methods (section 3.5.2.2)
Correspondence methods (section 3.5.2.3)
Warping methods
DID methods
Parameter methods
Local optical flow methods
Matching methods

Figure 3.23.: Categorization of local visual homing methods. After Möller and Vardy [452].

mobile robots relying on omnidirectional vision is identical to local visual homing operating on a feature-based representation of places.

3.5.2. Literature Review on Local Visual Homing

Over the years, a large number of methods for local visual homing was proposed. This review briefly covers the field by categorizing the different approaches following the categorization proposed by Möller and Vardy [452]. For more detailed reviews, please refer to [189, 447, 451, 452, 642, 660, 718, 719]. Beyond that, the review briefly documents Lorenz Hillen's research on local visual homing neither considered in this dissertation nor published so far. This includes the technical reports [213, 217] by Lorenz Hillen and supervised student projects [37, 328, 667, 669].

3.5.2.1. Depth-Based vs. Intensity-Based Homing

According to Möller and Vardy [452], local visual homing methods can be partitioned into methods relying on *depth information* and methods relying on *intensity information* (figure 3.23). Methods operating on *depth information* (figure 3.24) derive information about the distance of visible objects by using omnidirectional stereo setups (section 3.2.4.1). Deriving depth information requires more effort regarding the used sensor or the applied processing steps than relying on intensity information. Due to relying on depth information, methods operating on depth information are invariant against changes of the illumination. Methods belonging to this class are described by [124, 610]. The former extends the concept of image warping (section 3.5.2.2) to the depth profile, the latter determines the home direction based on the distance to a set of visible objects or landmarks. Hence it is a correspondence method (section 3.5.2.3).

Intensity-based homing methods solely rely on image intensities and, in contrast to depth-based methods, do not require stereo sensors or special movement patterns. However, achieving illumination invariance is difficult and requires appropriate image processing and comparison methods (section 3.2.3.2 and chapter 5). Intensity-based homing methods can be partitioned into *holistic methods* (section 3.5.2.2) and *correspondence methods* (section 3.3.1.3).

3.5.2.2. Holistic Homing Methods

Holistic homing methods use the entire image rather than establishing correspondences between local features as is the case for correspondence methods (section 3.5.2.3). Depending on how the home vector is computed, holistic methods can be categorized into *warping methods*, *DID methods*, and *parameter methods.* The term "holistic homing" methods is used in a broader sense referring to

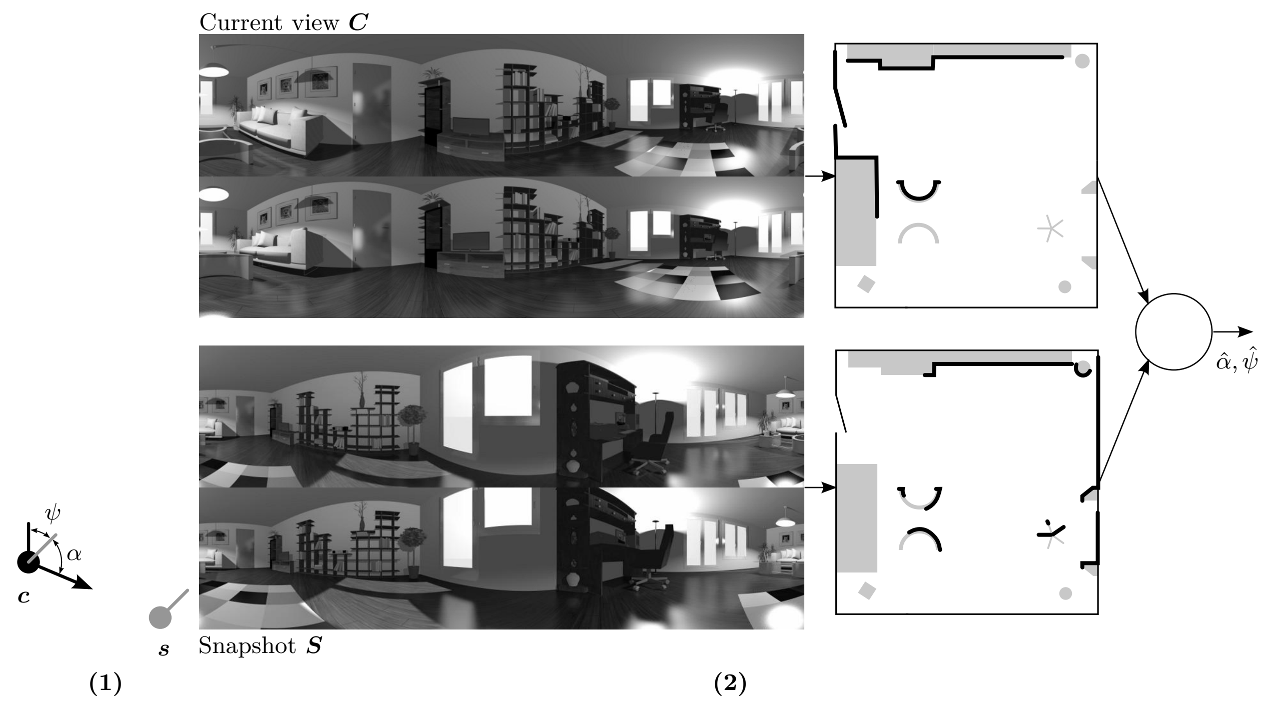

Current view $\boldsymbol{C}$
ψ
α
$\boldsymbol{c}$
$\boldsymbol{s}$
Snapshot $\boldsymbol{S}$
$\hat{\alpha}, \hat{\psi}$
(1)
(2)

Figure 3.24.: Depth-based local visual homing. Subfigure (1): situation in space; subfigure (2): sketch of the algorithm. The robot is located at position $\boldsymbol{c}$ and is supposed to return to position $\boldsymbol{s}$; its orientations at $\boldsymbol{c}$ and $\boldsymbol{s}$ are depicted by a solid lines. The angles α and ψ denote the true home direction and the true compass shift between the two robot poses. Depth based homing methods derive a distance profile (thick black lines; here created manually, the footprint is depicted for visualization only). The distance profile is computed from omnidirectional stereo information. In the depicted case, an omnidirectional stereo setup was used, which acquires two views at identical position in space but at different height above the ground (figure 3.18.1). This distance profile is by the homing algorithm (circle) used to compute estimates $\hat{\alpha}$ and $\hat{\psi}$ of the home direction and compass shift, respectively. Depth-based homing can process arbitrarily aligned images.

homing methods operating on the entire image. It subsumes warping and DID methods deriving the home vector from the entire image thus relying on a holistic representation of places (section 3.3.1.1), and parameter methods relying on a signature-based place representation (section 3.3.1.2).

Warping Methods

Warping methods rely on a holistic representation of places (section 3.3.1.1) and distort the robot's current view according to simulated movement directions (figure 3.25). The home vector is derived by searching for the best match between the distorted images and the snapshot image with the matching quality being determined by computing an image dissimilarity measure. Warping methods have proven to be robust and accurate homing methods and do not require the application of an additional external compass. Except for the method by [456], they solve the homing problem from a rather technical perspective. The methods of this class were originally developed for one-dimensional intensity images [190]. Improvements operating on 1D images include the warping of Fourier signatures [611] and for depth-based methods the warping of depth-signatures [610]. An extension to two-dimensional images was proposed by [447], later on improved by [451] and reformulated to a biologically plausible homing method [456]. Warping methods are best suited for mapping methods relying on a holistic representation of places, i.e. for building *topo-metric maps with holistic place representation* or *purely topological maps with holistic place representation* (sections 3.6.3.2 and 3.6.4.1). Since warping methods exhibit a moderate computational complexity and can achieve accurate homing and compass estimates, they are well suited for cleaning-robot control. In the

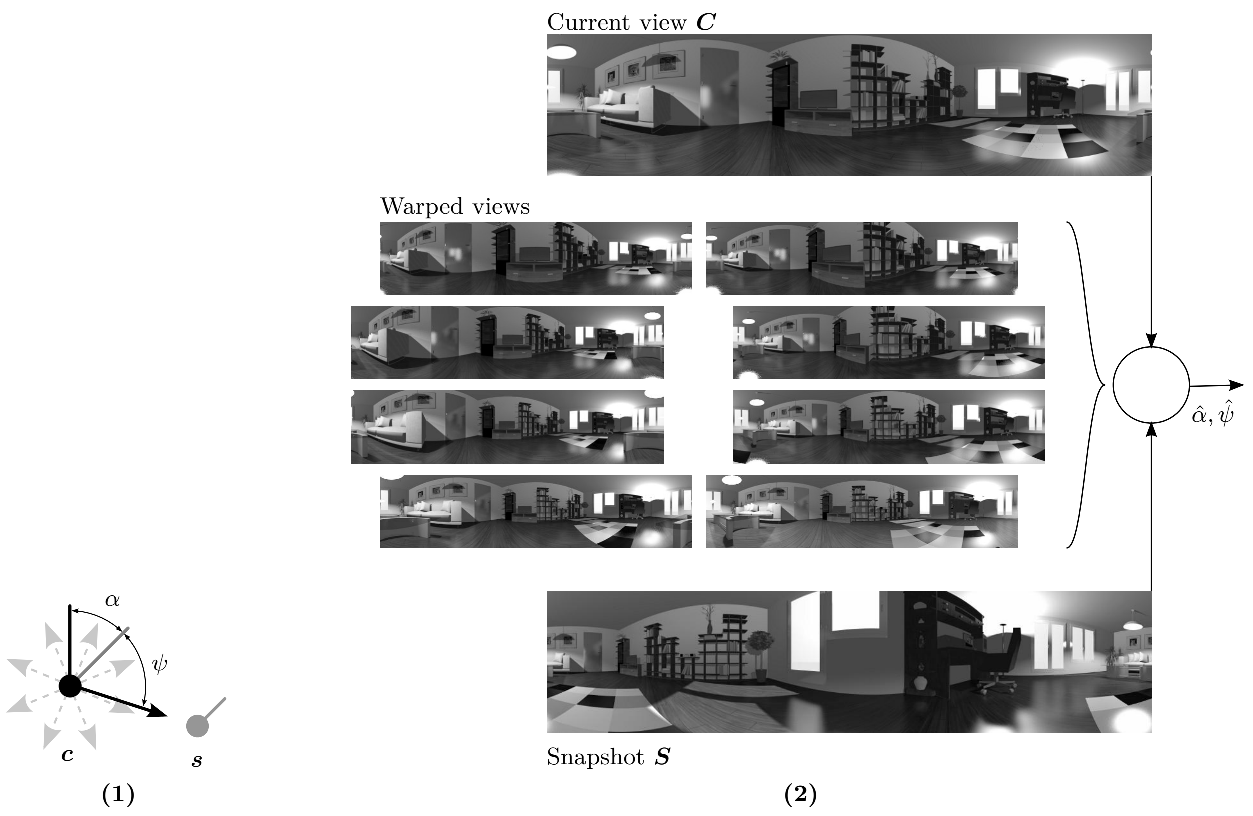

Current view $\boldsymbol{C}$
Warped views
$\hat{\alpha}, \hat{\psi}$
Snapshot $\boldsymbol{S}$
α
ψ
$\boldsymbol{c}$
$\boldsymbol{s}$
(1)
(2)

Figure 3.25.: Warping methods for local visual homing. Subfigure (1): situation in space; subfigure (2): sketch of the algorithm. The robot is located at position $\boldsymbol{c}$ and is supposed to return to position $\boldsymbol{s}$; its orientations at $\boldsymbol{c}$ and $\boldsymbol{s}$ are depicted by a solid lines. The angles α and ψ denote the true home direction and the true compass shift between the two robot poses. The current view $\boldsymbol{C}$ (acquired at position $\boldsymbol{c}$) is warped according to certain translational and rotational movement parameters (light-gray dashed arrows in subfigure (1)). The resulting warped views are compared to the snapshot $\boldsymbol{S}$ (acquired at $\boldsymbol{s}$). By an exhaustive search over movement parameters, the parameter combination resulting in the best match is determined and returned as home vector and compass estimates $\hat{\alpha}$ and $\hat{\psi}$, respectively. Warping methods can operate on arbitrarily aligned images.

context of this thesis we use the min-warping method proposed by [451] for mapping and trajectory control (section 4.4.2).

DID Methods

Like warping methods, DID (descent in image distances; figure 3.26) methods rely on a holistic representation of places (section 3.3.1.1). They achieve homing by gradient descent in the space of image distances computed from pixel-by-pixel comparisons between the current view and the snapshot. DID methods therefore rely on holistic representations of places (section 3.3.1.1). The homing method was originally proposed by Zeil, Hoffmann, and Chahl [718], and later on used by [13, 31, 103, 333, 404, 605, 612, 617, 618]. For computing the gradient, these methods have to apply exploratory test steps or special movement strategies. Among these methods, the papers by [31], [404], and [103, 605] describe homing behavior of desert ants, crickets, and rats, respectively. By predicting images into two orthogonal movement directions and deriving the gradient direction based on these predictions, the methods by [44, 354, 452, 454] circumvent the drawback of requiring exploratory test steps. These methods are rather technically-oriented DID methods. DID methods require snapshot and current view to be aligned w.r.t. a common direction or the application of an external compass method with holistic compass methods being best suited for this purpose (section 3.4.2.1). Regarding homing accuracy, DID-based methods do not reach

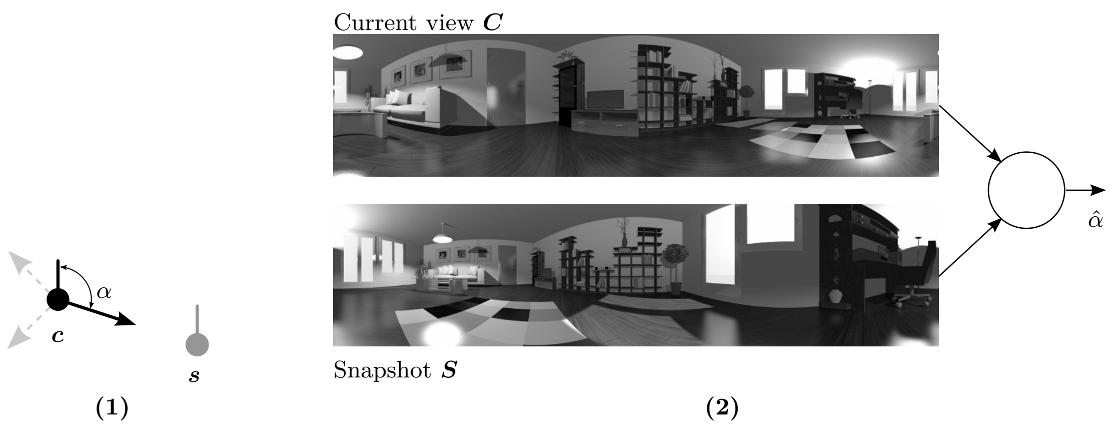


Figure 3.26.: DID methods for local visual homing. Subfigure (1): situation in space; subfigure (2): sketch of the algorithm. The robot is located at position $\boldsymbol{c}$ and is supposed to return to position $\boldsymbol{s}$. As DID methods assume images to be aligned w.r.t. a common reference direction, the robot's orientations (solid lines) at $\boldsymbol{c}$ and $\boldsymbol{s}$ are identical. The angle α denotes the true home vector pointing from $\boldsymbol{c}$ to $\boldsymbol{s}$. DID methods compare the current view $\boldsymbol{C}$ (acquired at $\boldsymbol{c}$) and the snapshot $\boldsymbol{S}$ (acquired at $\boldsymbol{s}$) by applying an image dissimilarity function. Homing is achieved by a gradient descent in the space of image dissimilarities. Computing the gradient direction (thick solid arrow; $\hat{\alpha}$) requires small test movements into two orthogonal directions (light-gray dashed arrows in subfigure (1)).

the accuracy of warping methods. They are computationally more demanding than parameter methods, but computing the home vector is more efficient than for warping methods. For mapping applications, DID methods are best applied for *topo-metric maps with holistic place representation* and *topological maps with holistic place representation* (sections 3.6.3.2 and 3.6.4.1). Due to their computational complexity, they would in principle be relevant for cleaning-robot control. However, we consider them to be not accurate enough for this particular task.

Parameter Methods

Parameter methods rely on a signature-based place representation and therefore extract a lower-dimensional description from the entire images (section 3.3.1.2). Homing is realized by a gradient descent minimizing the distance between the parameter signatures of current view and snapshot (figure 3.27). In principle, any signature reviewed in section 3.3.1.2 can be applied for homing. However, most signatures (e.g. statistical signatures, histogram-based signatures, and Fourier signatures; section 3.3.1.2) require translatory test movements for determining the gradient direction (dashed arrows in figure 3.27). In case the signatures are not independent of the robot's orientation, the images have to be aligned w.r.t. a common reference direction prior to computing the home-vector estimate. Rotation-invariant signatures can be applied without prior compass alignment, but only allow for estimating the home direction because orientation information is discarded (section 3.3.1.2). Comparison studies [177, 537] revealed that parameter methods do not achieve the homing accuracy of other classes, but are computationally very efficient. By applying the principle of image warping to Fourier signatures, a parameter-method which does not require test steps and which is capable of operating on arbitrarily aligned images can be obtained [611]. For hemispheric images (i.e. images with at least a hemispheric field of view; figure 3.5.2), spherical harmonics [146, 398], the spherical analogon to the Fourier transform in the plane, can be applied to derive signatures. From these signatures, the methods by [16, 136–138] derive estimates of the compass change and the home direction. The method by [402] applies similar concepts, but the approach is at the current point in time computationally not tractable for real-time control of robots.

Among the parameter methods which do not require exploratory test steps, the average landmark model (ALV) proposed by [356, 446] is most widely used. As signature, it uses the vector sum of

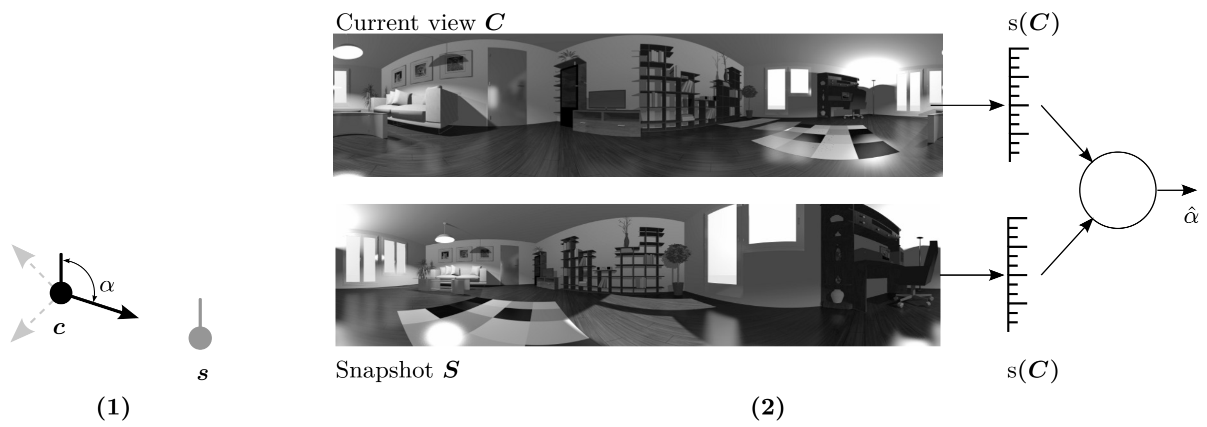


Figure 3.27.: Parameter methods for local visual homing. Subfigure (1): situation in space; subfigure (2): sketch of the algorithm. The robot is located at position $\boldsymbol{c}$ and is supposed to return to position $\boldsymbol{s}$. As parameter methods assume images to be aligned w.r.t. a common reference direction, the robot's orientations (solid lines) at $\boldsymbol{c}$ and $\boldsymbol{s}$ are identical. The angle α denotes the true home vector pointing from $\boldsymbol{c}$ to $\boldsymbol{s}$. By applying a signature function s, image signatures s($\boldsymbol{C}$) and s($\boldsymbol{S}$) are computed from the current view $\boldsymbol{C}$ (acquired at $\boldsymbol{c}$) and the snapshot $\boldsymbol{S}$ (acquired at $\boldsymbol{s}$). For homing, the signatures are compared instead of the images therefore allowing for efficient image comparisons. Homing is performed by a gradient descent in the space of parameter distances. For estimating the gradient (thick black arrow), small test movements into to orthogonal directions are required (light-gray dashed arrows in subfigure (1)). The direction of the gradient is returned as estimate $\hat{\alpha}$ of the home direction.

unit vectors pointing towards the landmarks —in this context visible objects— identified in the image. Thus, the signature is only two-dimensional. During period of time considered for this review, several extensions of the original method were proposed. The method by [267] circumvents the step of landmark identification and uses the center-of-gravity vector computed from image intensities instead of the landmark vector; the method by [244] uses color blobs as landmarks. In [238, 528], landmarks are identified by local image features thus extending the original method towards a correspondence-based homing method (section 3.5.2.3). The methods by [648, 649] and by [714, 715] use distance information to derive the signature and is therefore closely related to depth-based homing methods (section 3.5.2.1).

Due to their simplicity, ALV models were also applied to explain or model behavioral data of desert ants [31, 356] and crickets [404]. Bio-inspired applications include route following [592, 593] and docking [682, 683]. The drawback of ALV-based homing methods is that images need to be compass-aligned; only the methods by [714, 715] can cope with arbitrarily aligned images.

Parameter-based homing methods are closely related to *topo-metric maps with signature-based place representation* or *purely topological maps with signature-based place representation* (sections 3.6.3.2 and 3.6.4.1). Their computational simplicity makes them appealing for navigation of cleaning robots. However, we expect their precision to be not sufficient for this particular task.

3.5.2.3. Correspondence Methods

In contrast to holistic methods using the entire image to compute the home vector, correspondence methods establish correspondences for this purpose. Depending on how correspondences are established, one can distinguish *local optical flow methods* and *feature-matching methods.*

Local Optical Flow Methods

Correspondence-based homing methods establish correspondences implicitly by applying differential flow methods (figure 3.28). Since the optical flow (textbook: [305, 641], review: [30, 280]) is computed from the entire image, these methods rely on holistic place representations (section 3.3.1.1). Although

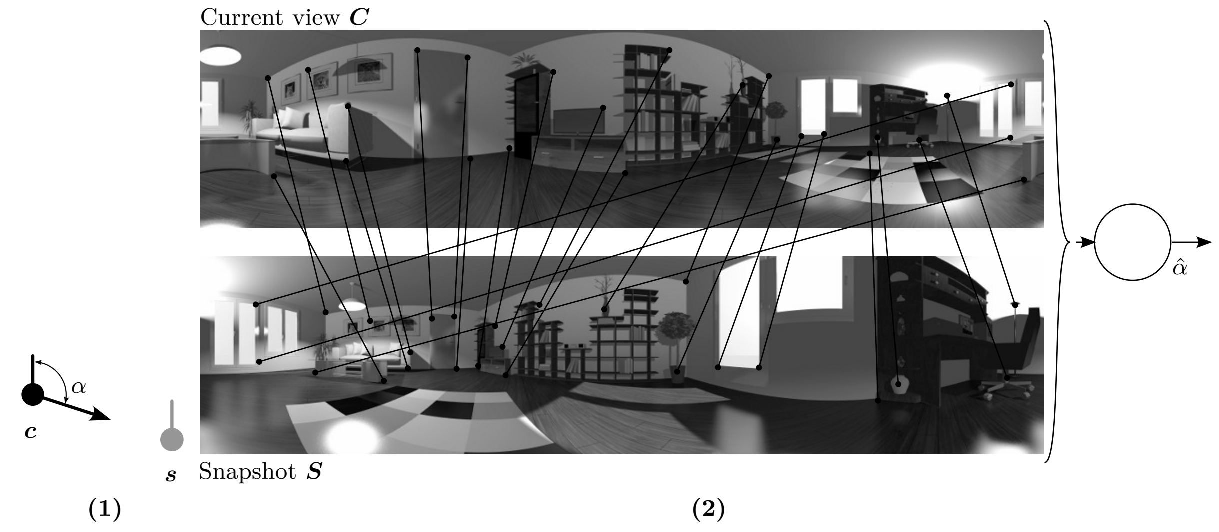


Figure 3.28.: Local optical flow methods for visual homing. Subfigure (1): situation in space; subfigure (2): sketch of the algorithm. The robot is located at position $\boldsymbol{c}$ and is supposed to return to position $\boldsymbol{s}$. As local optical flow methods assume images to be aligned w.r.t. a common reference direction, the robot's orientations (solid lines) at $\boldsymbol{c}$ and $\boldsymbol{s}$ are identical. The vector α denotes the true home vector pointing from $\boldsymbol{c}$ to $\boldsymbol{s}$. Local optical flow methods establish correspondences by computing the local optical flow between the current view $\boldsymbol{C}$ (acquired at $\boldsymbol{c}$) and the snapshot $\boldsymbol{S}$ (acquired at $\boldsymbol{s}$). Based on the correspondences, the home vector estimate $\hat{\alpha}$ is computed pointing from the current view position $\boldsymbol{c}$ to the snapshot position $\boldsymbol{s}$. In contrast to feature-matching methods for local visual homing (figure 3.29), local optical flow methods rely on a holistic representation of places (section 3.3.1.1).

differential flow methods assume snapshot and current view to be spatially close together —an assumption which is clearly violated if applying differential flow methods for homing—, these methods achieve a good, yet not perfect performance. Local optical flow methods require the application of an external compass to align images. For this purpose, holistic compass methods (section 3.4.2.1) are best suited. The homing methods by [663, 664] uses image processing operations closely related to local optical-flow computations, and the paper [660] is the first one to apply local optical flow methods. There, differential optical flow is computed by first- and second-order differential methods only capable of computing the *normal flow*. Such methods suffer from the aperture problem (textbook: [305]) and can only estimate the direction of the flow but not its length. By applying the Lucas-Kanade algorithm [390] and extensions thereof [61], the homing accuracy of the original methods could be considerably improved by Lorenz Hillen. The best results are obtained for a variant of the Lucas-Kanade algorithm relying on multi-scale flow computations with iterative refinement steps [61]. With this method, the homing accuracy could be considerably improved in comparison to the original method by [660]. The results of this line of research are documented in the technical report [213].

The method by [713] detects obstacles from computing local optical flow from consecutive frames of an image stream. It detects the robot's ground plane, computes the optical flow, and matches it with flow templates expected for the robot's current motion. By this means, the flow field is segmented into translational and rotational components both matching the expected motion and an unexpected part. The latter is due to flow resulting from obstacles. Although used for a different application and —more important— although an estimate of the robot's motion between snapshot and current view is not available for local visual homing, the concepts could also be interesting for local visual homing: home vector and possibly compass estimates could be computed from flow vectors explainable by translational and rotational flow components, respectively; the third category of flow components is not considered for deriving these estimates, but could be used to detect obstacles or dynamic scene changes.

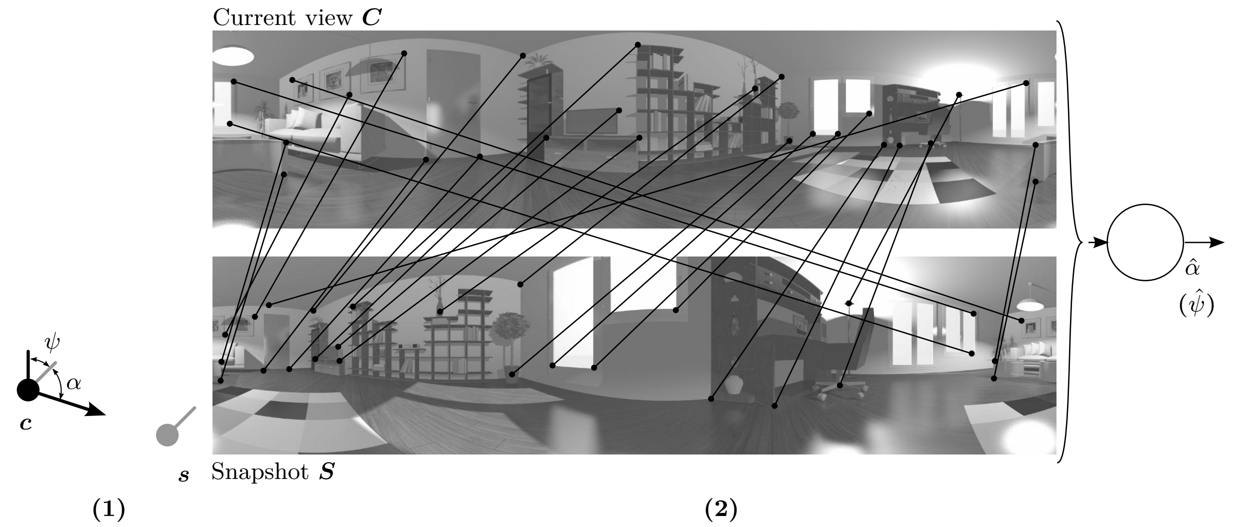


Figure 3.29.: Feature-matching methods for visual homing. Subfigure (1): situation in space; subfigure (2): sketch of the algorithm. The robot is located at position $\boldsymbol{c}$ and is supposed to return to position $\boldsymbol{s}$; its orientations at $\boldsymbol{c}$ and $\boldsymbol{s}$ are depicted by a solid lines. The angles α and ψ denote the true home direction and the true compass shift between the two robot poses. Feature-matching methods establish correspondences between the features of the current view $\boldsymbol{C}$ (acquired at $\boldsymbol{c}$) and of the snapshot (acquired at $\boldsymbol{s}$). From these correspondences, estimates $\hat{\alpha}$ and $\hat{\psi}$ of the home vector direction and the compass shift are derived. The depicted case shows a feature-matching method with feature preselection; most feature-matching can operate on arbitrarily aligned images. In contrast to local optical flow techniques (figure 3.28), correspondence methods rely on a feature-based representation of places (section 3.3.1.3) describing the image by a set of feature descriptors (and discarding the image content). For this reason, the current view and the snapshot are brightened in subfigure (2).

Another working direction includes computing the optical flow based on adaptive approaches to increase the method's robustness against illumination changes. In the diploma thesis by Dr. Matthias Behnisch [37], multi-layer perceptrons (MLP, textbook: [46, 148]) are used as adaptive function approximators to compute the direction of the flow vectors. Finding appropriate image-preprocessing methods and network topologies turned out to be rather subtle, and the homing accuracy could not be improved compared to methods based on the standard Lucas-Kanade algorithm [390].

Local optical flow methods achieve a similar performance than DID-based homing methods, to which they are closely related [452, 454]. For mapping applications, they are best applied with *topo-metric maps with holistic place representation* or with *purely topological maps with holistic place representation* (sections 3.6.3.2 and 3.6.4.1, respectively). For application to cleaning-robot control, homing methods based on optical flow do not offer particular advantages, but only the drawback that they have to rely on an external compass method. This was also the reason, why we did not further pursue this research direction and did not consider local optical flow methods for navigation of cleaning robots. Even though optical flow is a widely used cue in insect navigation ([250, 596, 597, 717] and section 3.5.3.1), we consider these methods to be technical solutions for the homing problem rather than realistic models of insect behavior.

Feature-Matching Methods

Matching methods explicitly solve the correspondence problem by matching features. In detail, these methods include the following four steps [451, 456]: (i) detection of points of interest, (ii) feature extraction, (iii) feature matching possibly including a removal of mismatches (iv) derivation of the home-vector estimate and depending on the used method of the compass estimate from the established correspondences (figure 3.29). Methods involving all four steps are referred to as *matching methods with feature preselection* [452]. In case the steps of detecting and extracting features —i.e. steps (i) and (ii)— are missing, the corresponding homing methods are referred to as *matching methods without feature preselection.* For comparing correspondence-based local visual homing methods, step (iv) is most essential. Thus, the following description will focus on this issue; steps (i) to (iii) are mostly independent of local visual homing and covered in section 3.3.1.3. Feature-matching methods are rather technical solutions to the local visual homing problem, and it is unlikely that similar mechanisms are used by animals or humans. The computational complexity of matching methods depends strongly on the used feature detection and description methods and on the approach for matching features in order to derive correspondences. The computational effort is larger than for DID and parameter-based homing methods (section 3.5.2.2) and should also be larger than that of local optical flow methods.

Matching Methods With Feature Preselection Matching methods with feature preselection include all four steps outlined above. Most methods establish correspondences in every homing step (figure 3.22), which is the bottleneck of correspondence-based homing methods. Only the methods by [14, 39, 173] reduce the computational effort by tracking features over time. Matching methods with feature preselection can be further categorized depending whether or not they estimate the essential matrix to derive derive estimates of the home vector and —in some cases— of the compass shift. The essential matrix describes the translation and rotation between camera poses up to scale.

Homing methods *relying on the essential matrix* can therefore cope with arbitrarily aligned images and can estimate both home vector and compass shift. The method by [408, 409, 411] uses special properties of the essential matrix to compensate for the rotation between images. Once the rotation is compensated, the translational component remains and the shift between correspondences is reduced to guide the robot towards the goal. The algorithm by [265, 434] establishes correspondences between straight lines and uses the matches to derive the control laws for guiding the robot towards its former pose when the snapshot was acquired..

In the context of trajectory-based SLAM (sections 3.6.1.2 and 3.6.3.2), homing methods are used to estimate spatial relations for accurate map-building [133, 134, 163, 164, 235–237]. These method focus on estimating the home direction and the compass shift, which is by all methods computed from factorizing the essential matrix into rotational and translational components. For details on the factorization please refer to the original literature (original paper: Nister [490]; tutorial [563]; textbook: [278]). The factorization can also be applied for feature-based place recognition (in case two places are identical, the translational component will be zero; section 3.3.1.1). Correspondence-based homing methods exploiting the epipolar constraint can fail if snapshot and current view are nearby, i.e. if the baseline between the two considered images is short. In such situations, the method by [385] can compute reliable estimates by deriving the homing direction and the compass estimate by establishing correspondences between vertical planes.

Closely related to methods computing the essential matrix are the works of [12, 36] which establish correspondences between three views (all other methods of this section only use two images for homing). Based on the correspondences between snapshot, current view, and the image acquired when homing started, the trifocal tensor is computed, and used to derive home vector and compass estimates. The trifocal tensor in three-view geometry corresponds to the essential matrix of two-view geometry (textbook: [278]).

Methods *not computing the essential matrix* instead exploit other properties of correspondences to derive estimates of the home vector and —in some cases— of the compass shift. These properties include their *position and shift* in the images, their *scale change*, or the *angle between matched features.* The group exploiting the correspondences' *position and shift* is formed by three very different methods by [57, 69, 560]. All three methods are capable of estimating both home direction and compass shift. The method by [560] applies the *vector-mapping* originally proposed by [660] relating each correspondence vector in the image with a home vector in space. The overall home vector is then computed by vector averaging. Prior to estimating the home vector, the method by [560] applies a feature-based compass to compensate for the robot's rotation between current view and snapshot (section 3.4.2.3). In [57], the problem of estimating the home-vector direction and the compass shift is formulated as maximum likelihood estimation. Given the set of correspondences, it searches for the most likely combination of homing and compass angles. The estimation is very efficient and can reduce the influence of mismatches e.g. caused by dynamic scene changes. The homing algorithm by [69] is the only homing method of this class operating on one-dimensional images and establishes matches keeping the ordering of matched features. Depending on the pattern how corresponding features are shifted between the images, home vector and compass estimate are obtained. This processing step makes the method closely related to warping methods (section 3.5.2.2 in particular our 2D warping methods [447, 451]).

The methods by [14, 38, 39, 374, 384] operate on the azimuthal *angle between matched features* and usually consider only a small number of landmarks (in this context visible objects) with some methods assuming known correspondences [38, 53, 384] or compass-aligned images [38, 384]. The approaches are often referred to as *bearing-only* methods —a term which we think is misleading because (i) these methods rely on the angles between matched features and (ii) all intensity-based correspondence methods use bearing-only information. The methods are mainly of theoretical interest for control-theoretical considerations such as proofs of convergence or stability (textbook: [143], tutorials: [303, 650]); only the methods by [14, 39, 374] were tested in real-robot experiments.

Homing based on the *scale change* between matches can be accomplished by moving towards increasing features and away from decreasing features [113, 114, 378, 379]. The technique is independent of the robot's orientation and can therefore process arbitrarily aligned images.

Because of representing places based on the visible features, correspondence methods with feature preselection are best used with *topo-metric maps with feature-based place representation* or with *purely topological maps with feature-based place representation* (sections 3.6.3.2 and 3.6.4.1). Feature-based homing methods would be applicable for navigation of autonomous cleaning robots because we expect them to be sufficiently accurate and computational efficient for cleaning robot control. Nevertheless, we prefer to apply our recent warping methods because we have a larger experience with this method.

Matching Methods Without Feature Preselection In contrast to matching methods with feature-preselection, these methods lack the steps of detecting and extracting features in the image (steps (i) and (ii) of above's list). Instead, they rely on the following matching process to establish correspondences. Feature descriptors are computed for every pixel of one image, usually the snapshot. To establish correspondences, feature descriptors computed for the current view are searched in the snapshot. The computational effort of the search process can be reduced by the following two means. First, correspondences are not established for every pixel of the current view but only for a subset of pixel positions (e.g. lying on a regular grid covering the entire image). Second, the search space in the current view is limited to an image region where matches are expected. Because of this search process, matching methods without feature preselection rely on a holistic representation of places (section 3.3.1.1). Based on the established correspondences, the home vector is derived (this step is again similar to correspondence methods with feature preselection). If not stated otherwise, the reviewed methods require current view and snapshot to be aligned w.r.t. a common reference direction.

The method described by [661, 665] establishes correspondences for matching scale-invariant feature descriptors. The algorithm proposed by [660] builds on the block-matching optical flow algorithm [35, 293] and establishes correspondences by matching image patches taken from the current view in a certain rectangular area of the snapshot. This original block-matching method was later on improved by Lorenz Hillen and by students being supervised by Lorenz Hillen. These improvements include (i) the features used for matching, (ii) the shape of the search region, (iii) the number and distribution of feature positions, and (iv) making the method independent of an external compass. The bachelor's thesis of Daniel Venjakob [667] uses techniques to approximate the image patches used for establishing correspondences by a lower-dimensional description. Due to only comparing lower-dimensional descriptors instead of image regions, this improvement allows for more efficient image comparisons and for comparing larger image regions. The tested image descriptors include principal component analysis (PCA, [471]), local jets [334], differential invariants [182, 576], and a descriptor based on the eigenvalues of the structure tensor [277]. Compared to the original method, jet features allow for more efficient image comparisons and did not decrease the homing accuracy. Furthermore, the work revealed that there is an optimal patch size. Both, for matching descriptors and for matching image patches, increasing the patch size up to an optimal value also increases the homing accuracy; for larger patches, it decreases again.

The technical report [217] by Lorenz Hillen describes a version of the original method referred to as flow-line matching. It uses image patches for establishing correspondences, but restricts the search space to optical flow lines. Optical flow lines are those lines in the image along which image features move when the robot moves [216, 334]. Thus, the method constrains the computed correspondences to image displacements which can occur under real robot movements. Although it does not achieve the performance of 2D-warping, flow-line matching performs considerably better than the original method by [660]. Among all our optical flow-based homing methods it currently achieves the best performance. Improvements could also be obtained with respect to computational complexity: by restricting the search space to the flow lines, the number of image comparisons mandatory to establish matches could be significantly reduced. In the bachelor's thesis of Björn Böttcher, flow-line matching was tested with different image preprocessing functions and different image dissimilarity function to increase the method's robustness against illumination changes (section 3.2.3.2). Among all tested methods, the combination of the contrast normalization by [612] and zero-mean normalized cross correleation (e.g. [225]) yielded the best results. However, if strong changes of the illumination occur, the performance is strongly decreased. Similar approaches for achieving robustness against illumination changes for loop-closure detection methods are described in (chapter 5).

In the course of the bachelor's project of Marcus Kesting [328], a variant of flow-line matching with feature preselection was implemented. For feature detection, the Harris corner detector was applied. Contrary to our expectations, the variant turned out to be less accurate than original flow-line matching. We additionally considered a variant of flow-line matching which tracks features over several homing steps. Thus, it circumvents the bottleneck of establishing correspondences in every homing step. At the current state, first results look promising, but the method still lacks a criterion for determining when feature tracking is no longer possible and correspondences have to be recomputed. Together with the robot's odometry, the tracking method allows for reconstructing the position of visible feature points in space. Thus, it opens a so far neglected approach scene reconstruction (e.g. for obstacle detection) and towards sparse model-based maps (section 3.6.3.1).

Another extension of the flow-line matching method integrated a visual compass into the method making it independent of an external compass. This extension was developed by Daniel Venjakob during his master's project [669]. Although testing several possibilities, flow-line matching with implicit compass does not reach the performance of 2D-warping (section 3.5.2.2) regarding homing accuracy and computational efficiency. This result is the main reason why the cleaning strategy described in chapter 4 of this thesis relies on 2D-warping instead of flow-line matching. Nevertheless, we could reveal that 2D-warping and flow-line matching are closely related: the equations used in the flow-line matching method for computing how image features move along flow lines and in the 2D-warping method for computing how image columns are moved and scaled are identical [669]. An interesting possibility of future work on optic-flow homing is the optic flow algorithm proposed by [375, 376]. It exploits constrains arising from considering the flow of two points which lie opposite to each other on the viewing sphere and allows for estimating both the robot's rotational and translational motion components.

3.5.3. Discussion of Local Visual Homing

The field of local visual homing has evolved in the context of biomimetic robotics (reviews: [189, 433, 642, 677, 678]). Thus, it is an interdisciplinary field between biology and engineering, and several algorithms for robot homing were proposed to model the insects' behavior. While the focus of section 3.5.2 was clearly on the technical side, we briefly cover the current state of biological and psychological homing research in section 3.5.3.1. Finally, we discuss the relevance of local visual homing for the work presented in this thesis (section 3.5.3.2).

3.5.3.1. Local Visual Homing and Spatial Cognition

Almost three decades have passed since the experiments by Cartwright and Collett [87] and Wehner and Räber [680] which lead to the snapshot hypothesis. The idea of the snapshot hypothesis is simple: places are characterized by the image as perceived at the corresponding place and homing is realized by bringing two images into accordance. However, it is until today not fully explained how places are characterized or how homing is achieved by animals (recent reviews: [246, 703, 717, 719]). In the following, we briefly recapitulate the current state of research related to the snapshot hypothesis for local visual homing in insects, rodents, and humans. Although birds use visual cues for returning to a nest or to a feeder (e.g. [298, 325, 326]), we are not aware of experiments with results being explained by the snapshot hypothesis.

Most research on local visual homing is done with social insects such as bees and ants gathering food and returning it to the nest. Such insects are referred to as *central-place forages*. In the experiments by [142], bees were able to approach a feeder position defined by three surrounding landmarks (in this context cylindrical objects). In contrast to the original experiments by [86, 87], landmarks were camouflaged (i.e. not visible if relying solely on image intensities), but visible if bees rely on the optical flow pattern (see [596, 597] for current reviews how bees use optical flow for flight behavior). Based on these findings, a matching of *flow templates* rather than traditional *snapshot* matching is proposed to explain the animals' behavior. For a more detailed discussion of static snapshots versus dynamic snapshots please refer to [141].

Homing experiments are also performed with desert ants (review: [679]). Recent results revealed the importance of the skyline panorama, i.e. the transition from sky to terrestrial objects or ground, for the homing behavior of ants. The experiments by [247] show that ants rely on the skyline panorama for navigation. Ants are trained to a certain skyline panorama at their food source. Under experimental conditions, ants are released in an arena with modified skyline panorama. This was possible due to using walls of variable height in order to rebuild the natural skyline. If this skyline panorama is rotated, the ants are misled. In another experiment, parts of the ants' view are obscured after accommodating the ants to a skyline panorama [248]. The results show that the lower portion of the panorama is sufficient and necessary for homing behavior. This portion of the field of view contains most parts of the skyline panorama. The study by [701] also modifies the ant's skyline panorama. It results are in line with the results reported by [247, 248]. The study of [31] compares parsimonious homing algorithms operating on (i) intensity images or (ii) on skyline information. Both image representations were computed from a computer simulation of a desert ant habitat. As the methods relying on skyline information yield more robust homing results, the authors suggest that ants rely on the skyline panorama. The skyline panorama could, for example, be extracted by color opponency mechanisms exploiting the spectral contrast between UV and green [337, 455, 537, 632]. Due to relying on UV information, they can only be applied for outdoor navigation.

Besides evidence on the information used by ants, a new line of homing models is proposed by

[245]. Instead of storing a single snapshot characterizing the goal, the authors suggest to store a set of snapshots acquired in close vicinity of the goal while pointing towards the goal. While approaching the home, the agent aligns with the best matching goal snapshot. This simple behavior is sufficient for homing and can also be motivated by learning walks and scanning behavior.

Research on local visual homing in rats is frequently related to the classical Morris-water-maze experiment [463, 464]. Animals are released in a tank filled with milky water and have to find a platform hidden underneath the water surface. As rats dislike swimming, resting on the platform seems to be a reward. The results of the experiments described by [272, 273] can be explained by the rats relying on nearby visible objects or cues as landmarks. However, it remains unclear (i) if the animals rely solely on visual cues and (ii) to which extent they rely on snapshot matching and not on other strategies using the visible landmarks. Another subbranch of research on homing in rats is influenced by the seminal experiment by [100]. Rats are trained to a feeder in a corner of a rectangular arena with only one wall painted in a different color. During experiments, rats supposed to find the correct corner confuse it with the diagonally opposite corner. Based on these results the authors concluded that rats use a geometrical module (review: [101]) deriving geometrical information about the rat's environment. Thus, rats were supposed to rely on geometrical information rather than on snapshot matching. Only recently, the experiments were repeated in a simulation study [103, 605] revealing that the results could also be explained by snapshot matching of images preprocessed by an edge detector. These results suggest that rats could use a representation of places closely related to holistic place representations (section 3.3.1.1).

Humans are in principle able to solve a local visual homing task in a virtual environment [227]. However, the relevance of the snapshot hypothesis and visual homing in human's everyday life is unclear, in particular because many other visual cues and navigation strategies exist. For details please refer to the reviews by [403, 690, 691].

All experiments reviewed above describe *local homing* with a homing range up to several meters. This contrasts *long-distance homing* with homing ranges up to thousands of kilometers. Well-known examples of such homing behavior include bird migration, pigeon homing, migration of monarch butterflies, or salmon migration (reviews: [64, 287]). Due to covering long distances, animals cannot rely on vision as main sensor modality for guidance towards the goal. They rather use magnetic compasses (birds: [45, 420, 466, 692, 693, 716], monarch butterflies: [534], sea turtles: [383]), sun compasses (birds: [467, 468, 693], monarch butterflies: [534]), celestial cues (birds: [629, 693]), or olfaction (birds: [204, 205, 420], salmon: [599, 645]).

3.5.3.2. Application of Local Visual Homing in This Work

Local visual homing was originally proposed as a method for guiding a robot back to a previously visited place. It is strongly influenced by the snapshot hypothesis of insect navigation, and several homing methods were developed as models to explain the insects' behavior. These methods are of particular interest, because they operate on small-sized images with a resolution of approximately 1° per pixel and do not incorporate computationally demanding processing steps (review: [453]). We therefore consider local visual homing methods to be suitable for navigation of autonomous floor-cleaning robots. Among all homing methods developed in our research group, the variant of 2D warping referred to as min-warping [451], offers a good homing accuracy while being computationally efficient (see section 4.4.2 for details on this method). The drawback of local visual homing methods is that their navigation range is restricted to the catchment area. By integrating several snapshot positions into a map, this drawback can be resolved. Possible approaches which are in particular suited for the application with holistic homing methods are *topo-metric maps with holistic place representation* and *purely topological maps with holistic place representation* (sections 3.6.3.2 and 3.6.4.1, respectively).

3.6. Map-Based Navigation[8]

Map-based navigation forms the third level of the proposed navigation hierarchy and involves navigation methods incorporating three or more views (section 3.2.2 and figure 3.2). Similar to previous section, we define map-based navigation (section 3.6.1), review related work (sections 3.6.3 and 3.6.4), and discuss relevant aspects (section 3.6.5).

3.6.1. Definition and Common Principles of Map-Based Navigation

Regarding the hierarchy of navigation methods introduced in section 3.2.2, map-based navigation methods form the third level incorporating three or more known positions. By these methods, several known places together with their spatial interrelations or positions in space are stored in a *map* of the robot's environment. How the map is built and how it represents the information about the robot's environment strongly depends on the used visual information (range or intensity information), on the representation of places (i.e. holistic, signature-based, or feature-based; section 3.3), on the navigation methods applied at lower levels (sections 3.3 to 3.5), and, of course, on the robot's task. Independent of the map's type, the amount of information stored in the map is only limited by the robot's storage capacity and computational power. Nevertheless, compact representations of space are preferable [74]. To accomplish reliable map-based navigation, the map has to be consistent, i.e. it has to correctly represent the robot's environment. For this purpose, the navigation method has to fulfill the following four prerequisites introduced in section 3.2.3 (see also [74, 99, 175, 432]): (i) robustness against perceptual aliasing (i.e. the capability to distinguish places with identical visual appearance; section 3.2.3.1), (ii) robustness against perceptual variability (i.e. robustness against illumination changes and changes of the scene; section 3.2.3.2), (iii) correct sensor data integration (i.e. the correct integration of information accumulated over time into the map; section 3.2.3.4), and (iv) correct place recognition (or loop-closure detection; i.e. the correct detection whether or not the robot's current position is already stored in the map in order to avoid inconsistent maps; section 3.2.3.3).

3.6.1.1. Basic Categorization of Mapping Methods

Although dating back to the beginnings of mobile-robot research, research on map-based navigation methods is still a vivid and only partially solved research domain. During the last decade, a substantial number of original research papers about building and using maps based on omnidirectional image information were published. However, the field lacks a state-of-the art review providing a sound categorization of map-based navigation methods relying on omnidirectional visual navigation. We therefore propose such a categorization (figure 3.30) tailored to the specifics of omnidirectional vision with a focus on different types of maps, i.e. on the spatial representations of the robot's environment built by processing the available image information. We are of the opinion that a categorization based on the spatial representation is best suited for putting our navigation methods in the context of related work. However, different but equally valid categorizations would result from focusing on different aspects of map-based navigation.

Approaches to mobile-robot mapping can be partitioned into *quantitative* and *qualitative* methods (figure 3.30). *Quantitative maps* (section 3.6.3) include an estimate of the robot's position with respect to an external frame of reference and, in some cases, also a position estimate of the visible features. In contrast, *qualitative maps* (section 3.6.4) do not contain metrical position information, but rather model spatial interrelations and characterize places by the sensory information, i.e. the camera image, perceived at the place.

[8]An earlier and much shorter version of this section was published in section 2 of our recent journal publication Gerstmayr-Hillen et al. [222].

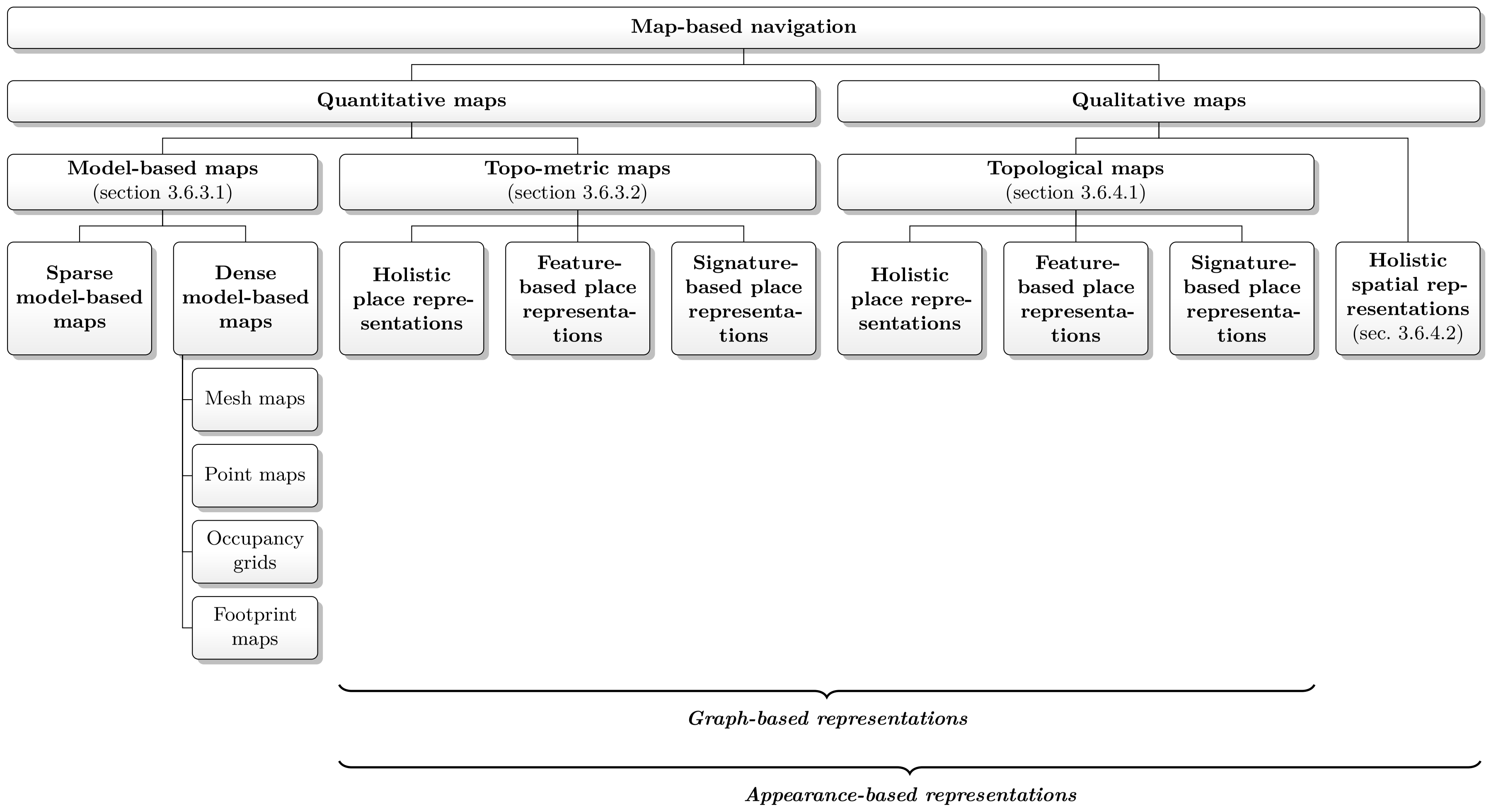

Map-based navigation
Quantitative maps
Qualitative maps
Model-based maps (section 3.6.3.1)
Topo-metric maps (section 3.6.3.2)
Topological maps (section 3.6.4.1)
Sparse model-based maps
Dense model-based maps
Holistic place representations
Feature-based place representations
Signature-based place representations
Holistic place representations
Feature-based place representations
Signature-based place representations
Holistic spatial representations (sec. 3.6.4.2)
Mesh maps
Point maps
Occupancy grids
Footprint maps
Graph-based representations
Appearance-based representations

Figure 3.30.: Categorization of spatial representations for map-based navigation methods. *Graph-based representations* and *appearance-based representations* embrace maps over different branches of the taxonomy. The former group subsumes methods relying on a graph structure for representing the robot's environment. The latter contains all methods which solely operate on image information without estimating the positions of visible features w.r.t. world coordinates.

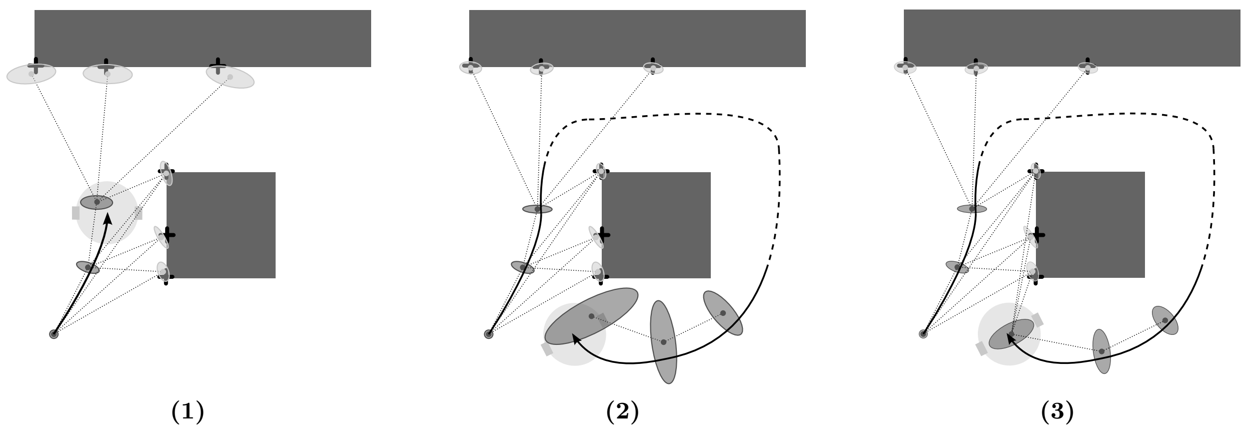


Figure 3.31.: Principles of feature-based simultaneous localization and mapping (SLAM). While moving, the robot continuously senses features in its environment (black crosses). Instead of mapping every measurement (dotted lines), the currently perceived features are integrated into the existing map. The map consists of position estimates of the perceived features (light gray filled circles) and the robot positions (dark gray filled circles) together with their uncertainties (ellipses). Subfigure (1): features perceived for the first time (elongated obstacle) are added to the map with position estimates deviating strongly from the true feature position and with a relatively large uncertainty. By fusing several measurements over time, the estimates become more accurate and the corresponding uncertainties decrease (squared obstacle). Based on the updated map, an estimate of the robot's current position is derived. Thus, the robot is concurrently localized within the map while the map is updated. Subfigure (2): over time, the estimates of the robot's positions drift, and the uncertainties accumulate because the robot does not perceive features stored in the map. Subfigure (3): as soon as the robot closes the loop and perceives known features, the relatively small uncertainty of these features is used to correct the current position estimate and to reduce the corresponding uncertainty. Extended after Thrun and Leonard [631].

Besides this basic partition of mapping methods, the terms *graph-based representation* and *appearance-based representation* embrace certain types of quantitative and qualitative maps (figure 3.30). *Graph-based representations* subsume *topo-metric maps* (section 3.6.3.2) and *topological maps* (section 3.6.4.1). Both types of map rely on a graph to represent the robot's environment. Graph nodes represent known positions in space, and two nodes are linked if the places they represent are directly reachable from each other. Each node has an omnidirectional image attached, which was acquired at the position represented by the corresponding node. In case an additional position estimate is attached to the place node (see section 3.3.2.1 for such estimation techniques), the map belongs to the class of *topo-metric maps*; otherwise, it is a purely *topological map*. *Appearance-based maps* embrace *topo-metric maps* (section 3.6.3.2), *topological maps* (section 3.6.4.1), and *holistic spatial representations* (section 3.6.4.2). Such mapping methods operate solely on image information and avoid to estimate the spatial positions of features w.r.t. world coordinates (section 3.2.2). Before describing the different approaches to mapping based on omnidirectional images (sections 3.6.3 and 3.6.4), we first introduce common tasks building or operating on maps (section 3.6.1.2) and comment on hierarchical maps (section 3.6.2).

3.6.1.2. Common Applications Related to Mapping

Nearly all complex applications of mobile robots require map-based navigation methods building a map of the robot's environment or using an a-priori known map. In the following, common building blocks of more complex applications are defined and their relevance for navigation strategies of autonomous floor-cleaning robots is discussed:

- *Map building*, or simply *mapping*, is the process of building a map as representation of the robot's workspace based on the robot's sensor data (reviews: [74, 432], textbooks: [110, 586, 630]). Mapping methods can be further separated into *online methods* building the map while the robot is navigating and into *offline methods* computing the map after the robot has explored its workspace and all visual information used for mapping is available. For all other tasks given in this list, reliable mapping is an essential prerequisite. A map is also the core of any navigation strategy for any autonomous cleaning robot, which systematically covers the workspace (section 2.3). To make the robot usable "out of the box", the map has to be built by online mapping methods.

- *Localization* is the problem of determining the robot's current position in its environment (reviews: [74, 175], textbooks: [110, 586, 630]). This involves a search for the position (either a discrete place or a continuous coordinate) for which the current sensor data best matches the map. How this search is performed depends on the used spatial representation. Localization algorithms can be partitioned into *local* and *global localization*: the former sequentially tracks an initial estimate of the robot's position whereas the latter determines the position without any prior knowledge. *Global localization* is also referred to as kidnapped-robot problem [586]. For an autonomous cleaning robot, accurate navigation is essential for (i) consistent mapping, i.e. for maps which correctly represent the robot's environment, (ii) reliably detecting cleaned areas by means of loop-closure detection (section 3.2.3.3), and (iii) for solving the kidnapped robot problem if the robot is displaced by the user.

- *Simultaneous localization and mapping* (SLAM) embraces navigation strategies which update the robot's map while concurrently localizing the robot in the map (reviews: [25, 99, 130, 150, 194, 631], textbooks [110, 586, 630]). Such methods arose from the tight coupling of *map building* and *localization.* While the former requires an estimate of the robot's position the latter requires a correct map of the robot's environment. SLAM algorithms involve a subsequent improvement of initial position estimates whenever new information about the robot's environment becomes available, e.g. after closing loops (figure 3.31). Traditional visual SLAM methods are closely related to *structure from motion* techniques (reviews: [130, 640], textbook: [278]) and estimate the robot's and the features' positions w.r.t. world coordinates. They are referred to as *feature-based SLAM* methods (section 3.6.3.1). *Trajectory-based SLAM* methods do not estimate the features' positions in the world, but only the current robot position and the series of former robot positions (section 3.6.3.2). Approaches to *topological SLAM* are qualitative methods and hence do not estimate positions at all. They rather aim at finding the most likely spatial interrelations of places (i.e. the topology, section 3.6.4.1). Our mapping method (chapter 4) concurrently extends the map while keeping track of the robot's current position. However, at the current state of our work, we avoid the computationally demanding step of subsequent position corrections inherent to SLAM methods and aim at covering the robot's workspace by a set of locally consistent segments. For a detailed discussion of this issue, please refer to section 4.6.1. Nevertheless, SLAM methods are applied by other research groups working in the field of cleaning-robot navigation (section 2.2.2), and it is likely that some commercially available robots rely on SLAM (section 2.2.1).

- *Visual odometry* is the process of visual ego-motion estimation and of integrating these estimates over time to obtain an estimate of the robot's current position (reviews: [191, 563]). The term was chosen to emphasize the similarity to wheel-based odometry, which is —due to influences such as wheel slippage— usually less accurate then visual methods. Visual odometry is closely related to *local localization.* Algorithms for visual odometry incrementally integrate ego-motion estimates computed from consecutive camera images or apply visual SLAM techniques (see section 3.6.3.1 for examples). The former case is some sort of "visual dead-reckoning". The focus of the latter group of methods is on accurately estimating the robot's position and not on building a globally consistent map as is the case for standard SLAM methods. Although not considered in this thesis, visual odometry could be helpful for cleaning-robot navigation because of the following three reasons: (i) it can detect if the robot's wheels are spinning (in this case, the wheel odometry measures a movement of the robot but the visual information does not change), (ii) it is —in contrast to wheel odometry— not influenced by wheel slippage which is important for navigating on carpets, and (iii) relying solely on visual odometry is a means for avoiding the costs of wheel encoders.

- *Path planning* is the process of computing a safe path to a goal position based on the information stored in the map (review: [442]; textbooks: [110, 586]). The concrete steps involved in computing the path from the information stored in the map strongly depends on the used map. For moving along the path, *route following methods* are applied. For navigation of cleaning robots, path planning is applied prior to approaching uncleaned areas or the charging station.

- *Route-following* is used to guide the robot along a path computed by *path planning methods* or along a *route map* (sections 3.6.3.2 and 3.6.4.1). For *quantitative methods* (section 3.6.3), standard approaches for trajectory control making use of the robot's current position estimate can be applied (review: [462]; textbooks: [41, 586]). *Qualitative methods* (section 3.6.4) subdivide the route into several segments and navigate between intermediate goals by applying local navigation strategies such as visual homing (section 3.5). While following the route, the robot is restricted to the learned route, and navigation fails if this route is blocked. In the context of cleaning-robot navigation, route following methods are applied to return to the docking station or to approach the start position of a new cleaning segment.

- *Scene reconstruction* or *image-based rendering* methods construct a 3D computer graphics model from a set of (omnidirectional) images (textbooks: [186, 584, 616]). The resulting 3D models are often photo-realistic and allow for rendering of novel views of the robot's workspace, i.e. views taken from different camera poses than those used to gather the information for mapping. Methods for scene reconstruction are closely related to *feature-based SLAM.* Scene reconstruction is not relevant for the low-level aspects of cleaning robot control considered in this dissertation. However, such methods are relevant for high-level features visualizing the robot's environment (e.g. for an advanced user interface or to extend the application area towards surveillance tasks).

- *Biomimetic navigation* summarizes navigation methods from the field of biorobotics. This research domain brings together engineering and biology (or related disciplines such as psychology) offering natural sciences a tool to verify models in real-robot experiments and offering engineers a source of inspiration for efficient, robust and parsimonious solutions of technical problems (reviews: [433, 677, 678]). If the focus of these methods is on solving technical problems rather than on exactly mimicking neural information processing, they are usually referred to as *bio-inspired strategies.* Biomimetic and bio-inspired navigation methods can be divided into the following three groups: (i) *models for insect navigation* aim at explaining the navigation capabilities of ants, wasps, or bees (reviews: [596, 597, 717]), (ii) *place-cell models* describe the neural processes of the hippocampus involved in navigation of mammals (reviews: [4, 279, 465]), and (iii) *cognitive mapping methods* model the environment following the ideas of Edward Tolman (original work: Tolman [634], collection: [308]). Our navigation methods for cleaning-robot control can be considered to be bio-inspired, because two essential building blocks, namely topological mapping and local visual homing, are strongly influenced by the snapshot hypothesis of insect homing (section 3.5.1).

Following the reviews by [25, 99, 130, 150, 194, 432, 631] and textbooks by [110, 586, 630], one could also understand *loop-closure detection*, i.e. decision whether or not the robot already visited its current position, as a sub-task of a more complex mapping application. However, we consider

Table 3.1.: Spatial representations and long-range navigation tasks. A cell in the table is marked by a ✓ if we are aware of a paper solving the application with the corresponding type of map built from omnidirectional visual information. See section 3.6.1.2 for definition of the applications.

	Map											
	Model-based					Topo-metric			Purely topol.			
Application	Sparse model-based maps	Point maps	Mesh maps	Occupancy grids	Footprint maps	Holistic landmarks	Signature landmarks	Feature-based landmarks	Holistic landmarks	Signature landmarks	Feature-based landmarks	Holistic sp. representations
Mapping	—	✓	✓	✓	—	—	✓	✓	✓	—	✓	✓
Localization	—	✓	✓	—	✓	—	✓	✓	✓	✓	—	—
Feature-based SLAM	✓	—	—	—	—	—	—	—	—	—	—	—
Trajectory-based SLAM	—	—	—	—	—	✓	✓	✓	—	—	—	—
Topological SLAM	—	—	—	—	—	—	—	—	—	✓	✓	—
Visual odometry	✓	—	—	—	—	—	✓	—	—	—	—	—
Route following	—	—	—	✓	—	✓	—	—	✓	✓	—	✓
Scene reconstruction	—	✓	✓	—	—	—	—	—	—	—	—	—
Biomimetic mapping	—	—	—	—	—	—	—	✓	✓	—	✓	✓

loop-closure detection to be closely related to visual recognition of places and already discussed it in section 3.3.

It strongly depends on the used map how the tasks described above are solved by a specific navigation strategy. Some tasks suggest using a certain type of map or vice versa, whereas others tasks can be accomplished with arbitrary maps. An example for the former group is the tight coupling of scene reconstruction and dense model-based maps; examples of the latter include mapping and localization. These interdependencies between map types and mapping tasks are summarized in table 3.1.

3.6.2. Hierarchical Maps

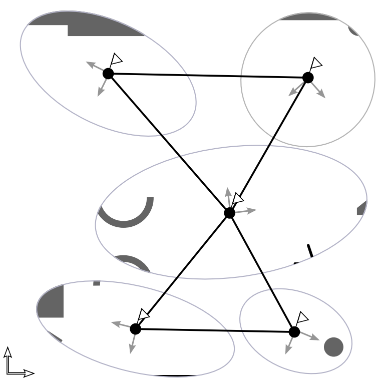
Independent on how position estimates are computed, these estimates have to be given with respect to some common frame of reference. Most of the quantitative methods proposed in the literature use a single frame of reference, usually referred to as world coordinate system. All estimates of the robot's position and —in case of model-based maps (section 3.6.3.1)— of feature positions are given with respect to this reference frame. For applications which build large or very dense maps, the computational effort to update the map or to operate on the map (e.g. for path planning) can grow considerably [76]. As operating on several smaller maps is computationally less demanding than operating on a single large-scale global map, several approaches segment the robot's entire workspace into multiple small submaps and link these submaps by an additional higher-level representation. Such methods are referred to as *hybrid maps* (theoretical aspects: [76]), as submap methods (review of corresponding non-visual SLAM techniques: [25]), as *hierarchical methods* (omnidirectional vision: [135, 407, 476], SLAM with range data: [165], SLAM with monocular stereo: [574]), or as *diktiometric methods* (early, non-visual work: [59, 160, 161, 582]). Here, we prefer *hierarchical maps* because this term expresses the segmentation of the robot's workspace into small submaps linked by a higher-level map.

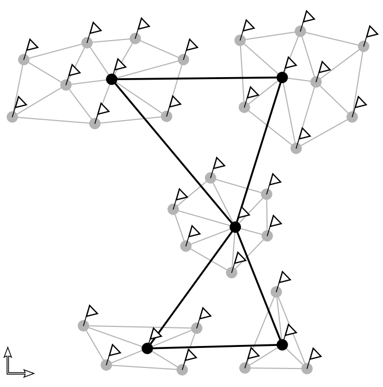
Table 3.2.: Overview of navigation strategies relying on hierarchical maps. The following abbreviations are used in the table: TM: topological map TMM: topo-metric map LM: landmark. References are subsumed by brackets if a single navigation method is described in several papers. The methods along the diagonal use the same *type* of map (but not the same maps) on both levels of the representation. Nevertheless, they clearly rely on a hierarchical representation of space and are therefore considered hybrid methods.

	High-level map				
Low-level map	Sparse model-based maps	TMM with feature-based LMs	TMM with signature LMs	TM with feature-based LMs	TM with signature LMs
Sparse model-based maps		[235–237] [628]	[210, 211] [474, 476, 477, 561]		
TMM with feature-based LMs	[424]	[55, 56] [54, 343, 725, 726]			
TM with feature-based LMs				[134] [475]	
TM with signature LMs					[601]

All approaches to hierarchical mapping have in common that the *low-level representation* and the *high-level representation* can be clearly separated. Usually, this separability results from using different types of maps for both levels. Table 3.2 summarizes and groups the works describing approaches to hierarchical mapping considered for this review. The table reveals three major groups of approaches to hierarchical mapping. The first group is depicted in figure 3.32.1 and uses a set of sparse-model based maps (i.e. a map with known feature position; section 3.6.3.1) as low-level map and combines these submaps by a topo-metric map (section 3.6.3.2). This group is formed by the methods of [210, 211, 235–237, 474, 476, 477, 561, 628]. The second group is depicted in figure 3.32.2 and relies on dense topo-metric map for the low-level and on a sparse topo-metric map for representing the higher level. The methods by [54–56, 343, 725, 726] belong to this class. The third group consisting of the approaches by [134, 475, 601] is identical to the second group except of using purely topological maps to represent the environment (figure 3.32.3). For the second and third group, a place node of the high-level map represents a region of the robot's workspace which is sampled by several place nodes of the low-level map. The work by [424] using a topo-metric map as low-level representation and a sparse model-based representation as high-level map has to be considered an exception.

Hierarchical maps allow for efficient localization. The detailed implementation of the localization strongly depends on the combination of maps used for representing the two levels of hierarchical maps. However, all localization methods proposed for hierarchical maps incorporate the following two steps. In a first step, the robot's current sensor data is compared to the high-level representation, and a best matching region is determined. For this region, the corresponding sub-map on the lower-level is consulted in the second step of the localization process. As the other submaps are not considered for localization, this procedure strongly decreases the computational complexity of localization. In contrast, localization in a single global map would also require to match the areas of the environment not considered for localization with an hierarchical map.


(1) Quantitative map as low-level representation in combination with a topo-metric map as high-level representation. Each local sparse model-based submap is depicted by a local coordinate system (light-gray arrows) together with the part of the workspace mapped by the corresponding submap (light-gray ellipse). As the position of visible features is known, mapping the robot's environment is possible (dark gray elements inside the ellipses). Local maps are linked by a topo-metric map (black graph).


(2) Topo-metric map as low-level representation (light-gray graphs) in combination with a more sparse topo-metric map as high-level representation (black graph). Nodes of both levels have a position estimate (flag) w.r.t. a common world coordinate system (open arrows) attached.

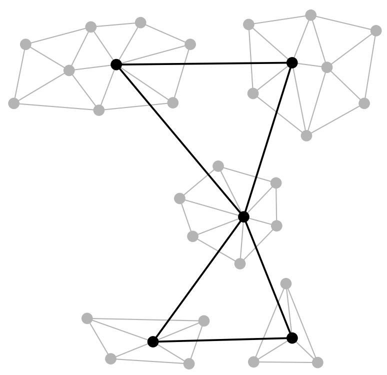

(3) Purely topological map as low-level representation (light-gray graphs) in combination with a more sparse topological map as high-level representation (black graph). None of the levels incorporates position information.

Figure 3.32.: Categories of hierarchical maps built from omnidirectional visual information. Subfigures (1) to (3) depict the most common cases as identified from table 3.2. Low-level representations are depicted in light-gray, high-level representations in black. If existing, the world coordinate system is depicted by open arrows, and position estimates w.r.t. this frame of reference are depicted by flags.

Hierarchical localization methods have to be distinguished from *staged localization* or *coarse-to-fine localization* strategies operating on a single global representation of space (e.g. [364, 366, 407, 473, 475]). For graph-based representations, localization involves computing the best match by comparing the robot's current sensor data with the sensor data attached to each place node of the map. Depending on the matching method and the size of the map, this process can be time-consuming. Coarse-to-fine localization strategies improve the localization performance by relying on the following two steps [364, 366, 407, 473, 475]: First, they compute a small subset of matching candidates by comparing the robot's current sensor data to *all* place nodes stored in the map, e.g. by applying an efficient matching method like histogram matching, [364, 366]. Then, for deriving the robot's position, the robot's current sensor data is compared to the matching candidates by applying a computationally more demanding but also more accurate matching method (usually based on establishing correspondences by feature matching). Thus, *staged localization methods* do not operate on two clearly separable and distinguishable representations of space as is the case for hierarchical methods. Such methods rather use two different approaches to recognize places —usually by combining signature-based and feature-based approaches (section 3.3.1.3).

For the review of map-based navigation methods presented in sections 3.6.3 and 3.6.4, we do not distinguish between *maps with a single frame of reference* and *hierarchical maps* because (i) the resulting taxonomy tree (figure 3.30) would be too branched and (ii) the spatial representations underlying both types of maps are the same. To this end, we only subdivide quantitative maps into *model-based maps* (section 3.6.3.1) and *topo-metric maps* (section 3.6.3.2).

Regarding vision-based navigation of cleaning robots, the decomposition of the robot's workspace into segments of parallel and meandering lanes suggests using hierarchical maps. In this case, each segment corresponds to a local submap, and local submaps are linked by a global representation of space. Since the task requires many operations on the level of a single segment, we see the main advantage of using a hierarchical representation in keeping the computational effort of map operations as low as possible. Methods for building and operating on such a hierarchical map are beyond the scope of this dissertation, but they will become relevant for future work (section 7.3.3).

3.6.3. Literature Review on Quantitative Maps

Quantitative methods —in the context of mapping also referred to as *geometrical methods*— build a geometrical representation of space and involve position information of the robot and —in some cases— of the visible features. Places are uniquely defined by their position with respect to some frame of reference [175] with the relation between visual information and a position in space depending strongly on the used type of map. Quantitative methods allow for approaching arbitrary positions or following trajectories by applying standard methods for trajectory control (review: [462]; textbook: [41, 586]). This allows for continuous and accurate navigation or localization and contrasts the standard usage of purely topological maps, which are usually applied for coarse navigation, i.e. for less precise or less accurate navigation (section 3.6.4.1). The available position information can also be used to visualize the maps. However, to estimate and to update the robot's position computational effort is required which is avoided by qualitative mapping methods (section 3.6.4). Quantitative maps can be further divided into *model-based maps* (section 3.6.3.1) and *topo-metric maps* (section 3.6.3.2). These two subgroups will be described in the following.

3.6.3.1. Model-Based Maps

Model-based maps are detailed and geometrical representations of the robot's environment which contain the positions of the mapped features w.r.t. an external reference frame. Hence, they rely on a feature-based representation of places (section 3.3.1.3). As the position of visible features, which are in the context of model-basd maps used as landmarks, in the world is estimated, mapping methods of this class are closely related to structure-from-motion methods which recover a model of the robot's environment by fusing visual information (review: [130]). Model-based maps are two- or three-dimensional maps which represent the robot's workspace by a dense set of geometrical primitives or by a sparse set of image features [237]. These two categories are referred to as *dense model-based maps* (section 3.6.3.1) and *sparse model-based maps* (section 3.6.3.1).

Both categories have in common that the spatial position of the landmarks is estimated. In order to update the map or in order to localize the robot within the map, the robot's current sensor data has to be transformed into a local map. This step requires a mathematical sensor model of the robot's internal and external sensors. Such sensor models also allow for fusing sensor information obtained by different modalities into a common map. In a second step, the obtained local map has to be matched with the existing global map in order to localize the robot or to update the map [139]. Due to the required sensor model and the matching method, model-based mapping methods are computationally more demanding than other long-range navigation methods.

Sparse Model-Based Maps

Sparse model-based maps represent the robot's environment by a set of landmarks with known positions in space (figure 3.33). A landmark is usually characterized by a feature descriptor computed at a point-of-interest and an attached estimate of the landmark's position in space. As the maps resulting from different types of features are very similar, we do not further subdivide this class. The available position information of the visible features can be directly used to visualize the

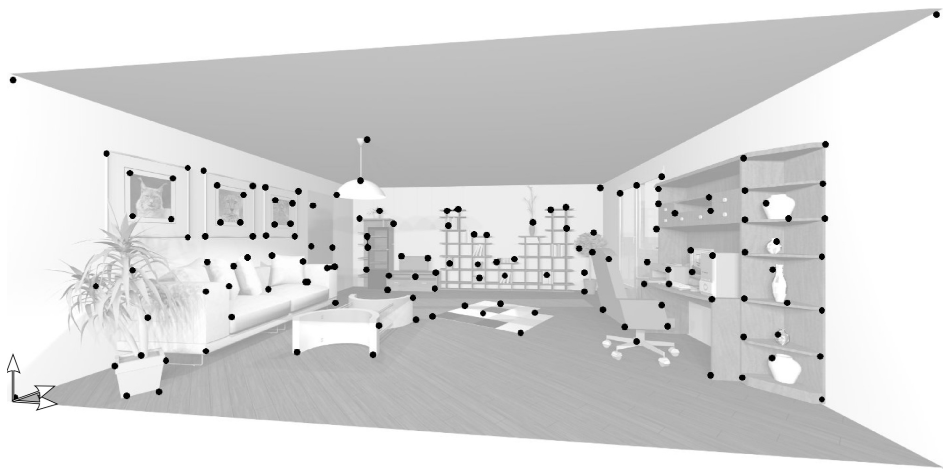


Figure 3.33.: Sparse model-based maps relying on point features as landmarks. For visualization purposes, the map (i.e. the set of features) is shown in combination with a rendered 3D view. The map's coordinate system is marked by arrows. The features were manually selected and not obtained by applying a feature-detection algorithm.

map. However, the resulting maps are difficult to interpret for human users —imagine figure 3.33 only as a set of points without the rendered view of the room. Sparse model-based maps are a memory-efficient representation of space but, because of the process of estimating and —in some cases— later on correcting the features' positions in space, can be computationally demanding. The maps scale linearly with the number of mapped features, but the effort required to maintain the map can have a worse complexity. For example, EKF-based approaches to SLAM have a complexity growing squared with the number of mapped landmarks [507]. In comparison to *point maps* (section 3.6.3.1), sparse model-based maps store only a small number of points detected by feature detectors (section 3.3.1.3) rather than modeling the robot's environment by a dense point cloud which consists of a large number of points.

Sparse model-based maps are built by feature-based SLAM techniques, by visual-odometry methods, and by some methods operating on hierarchical maps. The methods by [62, 206, 235, 237, 262, 292, 329, 331, 332, 364] are feature-based SLAM methods. Since the distance of visual features towards the camera cannot be estimated from a single monocular image (section 3.3.2.1), the methods which rely on a monocular omnidirectional image apply delayed updates [206, 262, 292, 329, 364]. In contrast, the methods by [62, 331, 332] operating on omnidirectional stereo images can perform undelayed updates. The methods can be further grouped depending on the used position estimation technique: [235, 237, 292, 329, 331, 364] apply a EKF framework, [206, 262, 332] apply FastSLAM (i.e. a Rao-Blackwellized particle filter), and [62, 147] apply optimization techniques. Among these methods, only the methods by [262, 364] detect loop closures, either by feature matching [262] or by histogram matching [364]. All other methods do not detect loop closures. Thus, only consecutive observations of a feature but not loop closures are used for subsequent position corrections.

The papers by [71, 82, 84, 322–324, 568, 569, 624] propose approaches to visual odometry. The algorithms estimate the motion between consecutive camera images and integrate the motion in order to obtain an estimate of the robot's current position and orientation. The cited methods differ in how (i) they incrementally estimate the robot's ego-motion and (ii) how these estimates are combined to a position estimate. To correctly estimate the robot's metrical position, the ego-motion estimation methods also have to recover the *length* of the translation between two views and not only its *direction* as is the case for local visual homing (section 3.5.1). However, the absolute length cannot be determined by purely vision-based methods because spatial relations are only defined up to scale; visual odometry methods solve this issue by (i) direct measurements (e.g. exploiting the known size of objects), (ii) motion constraints (e.g. non-holonomic constraints of the robot's motion), or (iii) sensor-data fusion with non-visual information (e.g. range information or inertial measurement data) [563]. These methods by [71, 82, 84, 322–324, 568, 569, 624] perform *visual dead reckoning* and —except for [568]— do not consider loop-closure detection. The method by [568] detects loop closures by image matching based on a vocabulary tree (section 3.3.1.3). In case loop closures are detected, all position estimates are corrected by applying an optimization technique.

The methods by [210, 211, 235–237, 476, 477, 561] operate on hierarchical maps. They all rely on a purely topological map as high-level representation and build local sparse model-based maps as low-level representation (figure 3.32.1). The submaps are used (i) to navigate accurately between two places nodes of the high-level map and meanwhile detecting obstacles [210, 211, 235–237] or (ii) to refine position estimates in a localization framework [476, 477, 561].

All methods reviewed in this section include a map-building stage. Although localization in the map is not covered, the robot's pose within the map is uniquely defined by the bearing to three or more mapped features. For trajectory control, standard methods operating on the robot's pose estimate can be applied (review: [462]; textbooks: [41, 586]). The method by [322–324] constructs a sparse model-based map as a basis to build a mesh map of the robot's environment (section 3.6.3.1).

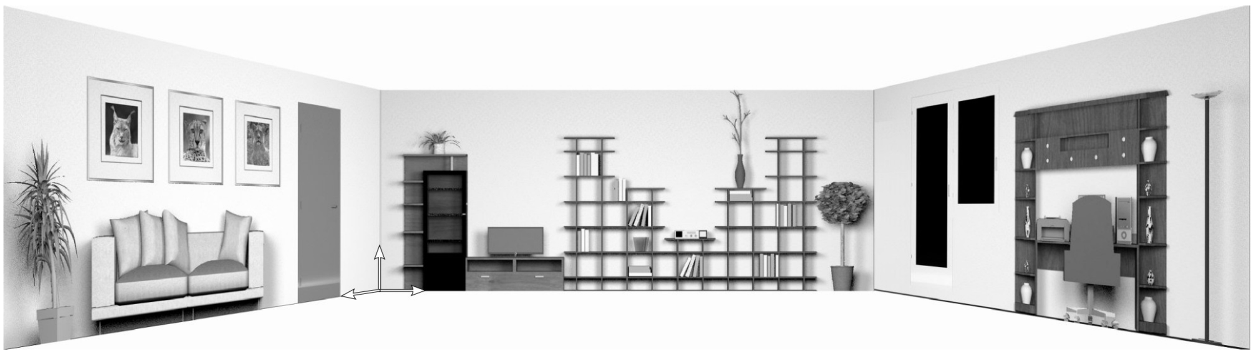
Sparse model-based maps are best suited for accurate navigation of mobile robots. The involved algorithms (i) to detect, describe, and match features and (ii) to estimate positions are well understood. Sparse model-based maps require considerable computational effort, but with modern computers, it is possible to compute these maps in real-time. With these properties, sparse model-based maps are also suitable for visual control of cleaning robots. The visual SLAM methods reviewed in section 2.2.2.2 are all feature-based SLAM methods building sparse model-based maps, and it is likely that some of the commercially available robots (section 2.2.1) also build sparse model-based maps. All these strategies rely on visual information obtained from a monocular directed camera (section 3.2.4.1). Even though these methods could be adopted for application with omnidirectional vision, we prefer appearance-based methods which avoid the computationally demanding step of estimating spatial positions of visible features.

Dense Model-based Maps

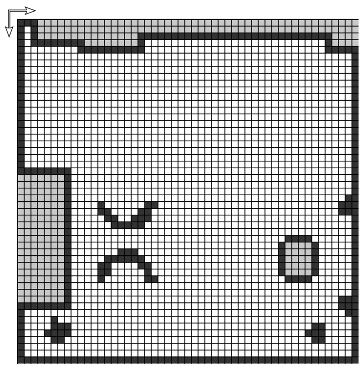
Dense model-based maps represent the robot's workspace by a set of geometric primitives (figure 3.34). Depending on the geometric primitives used to represent the environment, one can distinguish the following subtypes: (i) three-dimensional *mesh maps* relying on piecewise planar surface patches (figure 3.34.1), (ii) two- or three-dimensional *point maps* representing the environment by a dense point cloud[9], (iii) two- or three-dimensional *occupancy grids* formed by grid cells (figure 3.34.2), and (iv) two-dimensional *footprint maps* using lines to build a map similar to an architectural floor plan (figure 3.34.3). A similar categorization was proposed by Burgard and Hebert [74]. Since the positions of the geometric primitives which form the map are known, dense model-based maps can be visualized easily. To build and update sparse model-based maps computationally demanding processing steps can be involved. However, the computational complexity and memory requirements strongly depend on the type of map, on its spatial resolution, and on the size of the mapped area. . Due to their dense environmental representation, human users can easily operate on dense model-based maps. In the following, we will describe the four subtypes of dense model-based maps in more detail.

Mesh Maps Mesh maps represent the environment by a mesh, i.e. a set of three-dimensional surface patches, textured with image content in order to obtain a photo-realistic model of the environment

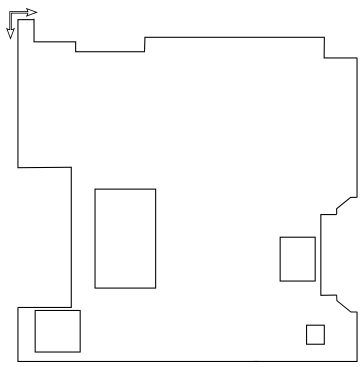
[9]We do not present point maps because they are very difficult to visualize with 3D-rendering techniques. They are similar to sparse model-based maps (figure 3.33) but model surfaces of the robot's environment by a dense point cloud (in contrast to a sparse set of interest points).



(1) Mesh map. The map consists of a set of piecewise planar surface patches.



(2) Occupancy grid. Depicted is a two-dimensional grid, but 3D grid cells are also possible.



(3) Footprint map. Footprint maps use lines to build a map similar to an architectural floor plan.

Figure 3.34.: Dense model-based maps represent the robot's environment by a set of geometric primitives. Depending on the used primitives, one can distinguish the following different subtypes depicted in subfigures (1) to (3). Another further subtype, namely point maps, are not included because they are difficult to visualize by 3D-rendering techniques. They are similar to sparse model-based maps (figure 3.33), but contain much more points to model the robot's environment. The maps' world coordinate system is depicted by arrows; the shown examples were manually drawn and not obtained from applying one of the reviewed algorithms.

(figure 3.34.1). Building mesh maps starts with an already existing map. This can be a sparse model-based map [83, 211, 322–324], a 3D occupancy grid [301, 444], a point map [369, 371], or a map built by fusing laser-range scans [42]. These maps are used for fitting surface patches in order to derive the mesh representation, which is in a second step textured with image content. Depending on the depth levels contained in the map, the resulting maps can be more detailed by modeling also obstacles or furniture [144, 301, 322, 437, 444] or more abstract by projecting obstacles or furniture to the walls of the workspace [42, 83, 202, 211, 322–324, 369, 371, 437]. The latter case is similar to the example depicted in figure 3.34.1. The cited papers all describe how the mesh map is built from an existing map. If considered at all, loop-closure detection is solved at the earlier mapping stage. Beyond mapping, the methods by [202, 444] propose localization methods based on the following two stages: first, position candidates are computed by range matching, which are in the second step refined by comparing visual information.

Mesh maps are best suited for post-processing an existing quantitative map to achieve a better visualization of the map and to simplify user interaction. However, for navigation tasks, they do not offer more or better information than the map they are built from. The map's level of detail can be adjusted depending on the requirements of the tasks and depending on the available computer hardware. As mesh maps are not better suited than other quantitative maps (in particular sparse model-based maps) for robot control, we do not consider them a good choice for the low-level aspects of cleaning robot control considered in this thesis. Regarding high-level aspects of cleaning-robot control, such maps could become interesting if a 3D visualization on an external display (such as a mobile phone or a computer monitor) is desirable. Possible tasks could include surveillance of the user's home or to guide the robot to a position by marking it on the map.

Point Maps Point maps represent the robot's environment by a dense two- or three-dimensional point cloud obtained from dense stereo matching between two or more omnidirectional images (section 3.3.2.1). These maps are related to sparse model-based maps (section 3.6.3.1) because the map contains a set of points with known position in space. However, the point maps differ from sparse model-based maps in the following three aspects: (i) point maps are formed by a larger number of points, (ii) the point cloud forms a dense model of the robot's environment, and (ii) the points do not correspond to local features detected in the image. By fitting surface patches to the point cloud, the original *point map* can be converted into a *mesh map*.

Most methods build 3D point maps [15, 70, 73, 180, 181, 344, 368–371]; only the method by [435, 436] builds a two-dimensional point map. The methods by [15, 70, 73, 344] compute point maps by dense stereo computations from a small number of images; the maps have a moderate resolution and cover a single room. Methods building large scale and high-resolution maps from many images include [368–371] for purely vision-based mapping and [180, 181] for fusing visual information and laser-range data. The resulting maps cover entire buildings or complete streets. The maps built by all methods of this class are used for visualization purposes but not for navigation relying on the built map. Thus, the robot is used for collecting the sensor data required to build the map but does not operate on the map.

Point maps are best suited for applications requiring a high-quality visualization or scene reconstruction from previously gathered sensor data. Since large and detailed maps can contain a huge amount of points, point maps are computationally very demanding and their applicability for real-time control of robots is limited. Therefore, we consider them to be not suited for the low-level aspects of cleaning robot control this thesis deals with. Point maps could be an interesting choice for high-level features which require a detailed visualization of the robot's environment. However, it is questionable whether or not such maps can be computed with a robot's on-board computer.

Occupancy Grids Occupancy grids were first described by Moravec and Elfes [461] and represent the robot's workspace by a discrete two- or three-dimensional grid (figure 3.34.2). Each grid cell represents the probability that the corresponding position in space is occupied (e.g. by an obstacle). Thus, free space usually has a probability close to zero (figure 3.34.2, white grid cells), obstacles receive probabilities close to one (black), and unknown or invisible areas typically have probability 0.5 (gray). Depending on the dimensionality of the grid cells used to represent the robot's environment, occupancy grids can be further categorized into *2D occupancy grids* [119, 443, 483, 485, 514, 515] and *3D occupancy grids* [301, 444, 527]. By increasing the spatial resolution of three-dimensional occupancy grids and by discarding grid cells which represent free space, a spatial representation similar to *point maps* is obtained. Regarding memory requirements, occupancy grids are less efficient than other methods because they also represent invisible and hence inaccessible areas of the robot's workspace and not only obstacle borders or free space.

Since occupancy grids were developed to fuse uncertain range information, the methods of this class rely on omnidirectional visual range measurements. The robot's current range measurement are matched against the existing map in order to estimate the robot's position. This *scan-matching* process also solves the loop-closure detection problem. Hence, mapping methods which build occupancy grids do not need an extra loop-closure detection based on *image comparisons* (section 3.3). In a second step, the grid probabilities are updated according the sensor data and the position estimate. Purely vision-based methods [119, 514, 515, 527] obtain the range profile

(i) from a single omnidirectional image by applying depth-from-elevation techniques [514, 515] (figure 3.18.3), (ii) from two omnidirectional images acquired with a omnidirectional stereo setup [527] (figure 3.18.1), or (iii) from two consecutive omnidirectional images acquired with a monocular omnidirectional vision setup [119] (figure 3.18.2). The method described by [443, 483, 485] fuses omnidirectional stereo information and range measurements obtained from a laser range finder.

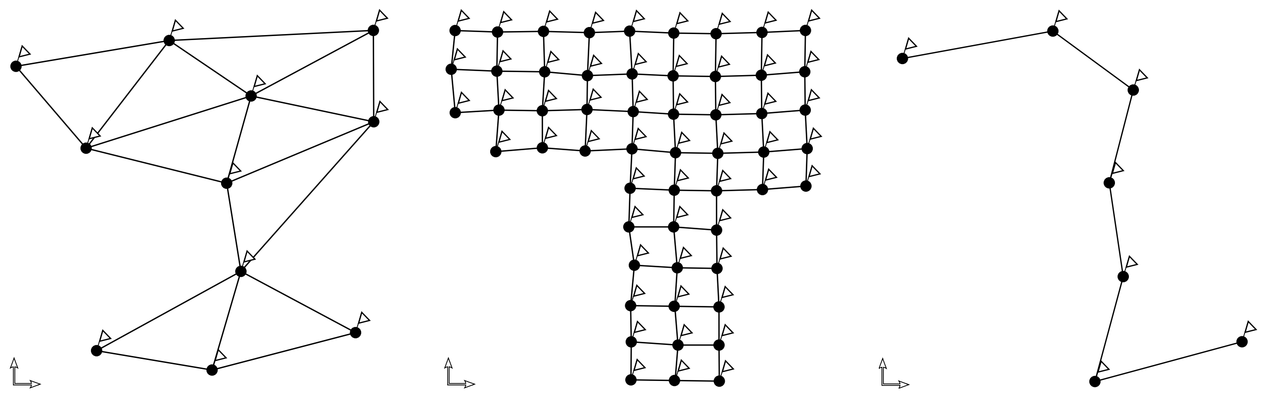
Most approaches of this domain solely describe the mapping process [119, 514, 515]; only the method by [443, 483, 485] also describes a path-following task. We are neither aware of papers that deal with visual localization in occupancy grids as a standalone problem (i.e. localization besides estimating the robot's position required for mapping) nor with subsequent map corrections based on omnidirectional visual information as they are usually pursued by SLAM algorithms. For navigation tasks which rely on occupancy maps, standard algorithms to plan paths and to avoid obstacle can be applied (textbooks: [110, 359]). Since the robot's pose is known, navigation strategies operating on occupancy grids can rely on standard techniques for trajectory control (review: [462]; textbooks: [41, 586]).

Occupancy grids are well suited if range information is available and —at least for large workspaces or high-resolution maps— if memory capacity is not critical. They allow for applying established standard algorithms for tasks such as obstacle avoidance or to plan and follow trajectories. Regarding the applicability for a cleaning robot, we currently see two drawbacks of occupancy grids: (i) they are memory-intensive representation (in particular if entire apartments have to be mapped at a fine spatial resolution) and (ii) we would have to compute range data from matching two consecutive images because our robot is not equipped with a sensor that provides range information. Coarse occupancy grids with a grid size being identical to the robot's size could be interesting for cleaning-robot control. They can be used to represent obstacles and the cleaned portions of the robot's workspace. The state of the latter cells could represent if the cell faces uncleaned areas or if it is completely surrounded by cleaned areas or obstacles. The video [I108] suggests that the Samsung robots (table 2.2) could apply such a map in conjunction with a feature-based SLAM method (section 2.2.2.2).

Footprint Maps Footprint maps represent the robot's environment by a set of two-dimensional lines which resembles an architectural floor plan (figure 3.34.3). Such maps are a complete and dense model of the robot's environment containing similar information than 2D occupancy grids but requiring less storage. Footprint maps are mostly used in the context of RoboCup, where the footprint represents the field markings of the playground [282, 297, 306, 422, 423]. Only the paper by [423] describes an application in an office environment using an architectural footprint of the building (like depicted in figure 3.34.3). The dense and complete model of the environment distinguishes footprint maps from *sparse model-based maps* (section 3.6.3.1) and *graph-based maps with holistic place representation* (sections 3.6.3.2 and 3.6.4.1). Such maps can rely on line features, but use only the most prominent line features detected by a point of interest detector (see the references given in figure 3.15). For sake of clarity, we therefore prefer *footprint maps* over *line maps* as proposed by [74].

All strategies relying on footprint maps operate on a-priori known maps and do not contain a mapping stage therefore avoiding both map-building and loop-closure detection. In the context of RoboCup, this assumption is reasonable because the size of the playground and the positions of the field markings are defined by the RoboCup rules. All methods of this class are applied for robot localization, either globally [297, 306] or incrementally by Monte-Carlo localization [282, 422, 423].

Footprint maps are best suited for navigation and localization in an environment, which can be modeled by a set of lines with known positions not changing over time. These maps are tailored to an environment model such as an architectural floor plan, which has to be installed on the robot prior to usage. Thus, its area of operation is restricted to the installed maps. As we are currently not



(1) Sparse topo-metric map. This is the most common type of topo-metric maps.

(2) Dense topo-metric map. Such maps sample the robot's environment more densely and often more regularly than sparse topo-metric maps.

(3) Topo-metric route map. Route maps are topo-metric maps which are not branched

Figure 3.35.: Types of topo-metric maps. Subfigures (1) to (3) depict sparse topo-metric maps, dense topo-metric maps, and topo-metric route maps, respectively. Places nodes consist of an image acquired at the corresponding position and a position estimate (flags) w.r.t. the reference coordinate system (open arrows). The exact spatial arrangement represented by the map can be visualized because the position of the place nodes is known (in contrast to purely topological maps). As the spatial positions of objects visible in the images are unknown, the robot's environment is not depicted. The purely topological counterparts of the maps depicted in subfigures (1) to (3) are visualized in figures 3.36.1 to 3.36.3.

aware of methods to extend and update footprint maps based on omnidirectional visual information, we consider sparse model-based maps the better choice if mapping is required for the robot's task. We are of the opinion that footprint maps are not suited for navigation of domestic cleaning robots because the user has to provide an environment model which we think is not desirable for household robots.

3.6.3.2. Topo-metric Maps

Topological maps with metrical information, or *topo-metric maps*, combine the advantages of both *quantitative* and *qualitative* maps in a spatial representation with a single frame of reference [432]. Like topological maps (section 3.6.4.1), they rely on a graph-based representation of the robot's environment (figure 3.35): places are nodes in the graph and are characterized by the raw camera image (or by information derived directly from the image) taken at that place in combination with an additional estimate of the robot's position at the time of image acquisition. Methods which rely on topo-metric maps are appearance-based methods because they do not estimate the spatial positions of visible features but solely rely on image-intensity information. Due to the graph-based representation, all places stored in the map are former robot positions, which are used as landmarks to derive spatial relations or to correct the map. Most methods estimate the robot's full pose, i.e. its position and orientation w.r.t. world coordinates. Some methods, such as the route-following method by [720] and the trajectory-controller proposed in chapter 4, only perform partial-pose estimation in order to avoid unnecessary computations.

Based on the maps' graph structure, topo-metric maps (and purely topological maps; section 3.6.4.1) can be categorized into *sparse topo-metric maps*, *dense topo-metric maps*, and *topo-metric route maps* (figure 3.35). The vast majority of navigation methods operate on sparse topo-metric maps (figure 3.35.1), which only add nodes at strategically important places with in the catchment area (section 3.5.1) of neighboring nodes. Keeping the map as sparse as possible is advantageous because it reduces the amount of memory required to store the map and the computational effort required to operate on the map. *Dense topo-metric maps* (figure 3.35.2) are usually applied if the robot's task requires a denser sampling of the workspace, i.e. to increase navigation accuracy. Dense topo-metric maps often have regular or grid-like structure [171, 172, 421, 505, 506, 538]. The graphs underlying both sparse and dense topo-metric maps are branched. *Topo-metric route maps* (figure 3.35.3) only represent a single route and the graphs used as representation are not branched. Another possible categorization of topo-metric maps is to group methods depending on the used place representation into *topo-metric maps with a holistic place representation*, *topo-metric maps with a signature-place based representation*, and *topo-metric maps with a feature-based place representation*. To avoid repeated descriptions, we do not further subdivide the remainder of this section but give a description of topo-metric maps which focuses on the various applications of such maps: map-building, localization, route following, navigation of a service robot in a do-it-yourself store, and cognitive modeling.

Map-Building

Building topo-metric maps can either be done *offline*, i.e. after the robot has explored its workspace and all images used for mapping are available, or *online*, i.e. while the robot explores the workspace. *Offline methods* for building dense topo-metric maps with a signature-based representation of places include [421, 505, 506]. Topo-metric maps can also be obtained by reintroducing distance information to a purely topological map followed by map relaxation. This approach is by pursued [268, 269] for a holistic place representation. The algorithm by [601] builds a hierarchical map relying on a sparse topo-metric representation for both the low-level and the high-level map. The method clusters the place nodes by local eigenspace methods and applies relaxation techniques between clusters (global level) and within each cluster (local level).

Most *online methods* for building topo-metric maps are trajectory-based SLAM methods. To optimize the position estimates attached to the place nodes of the map, these methods fuse odometry information with compass and bearing information obtained from local visual homing (section 3.5 or related ego-motion estimation techniques). Such methods were proposed for holistic [294, 441, 698], signature-based [261], and feature-based representation of places [5, 6, 34, 133–135, 163, 164, 366, 393–396, 551–554, 557–559]. The latter group of methods differ in the used features and the applied position estimation framework (section 3.3.2.1). As these differences do not have an influence on the resulting spatial representation, we do not further discuss differences between these methods. Loop-closure detection requires to recognize that the currently perceived image is identical to an image already stored in the map. Therefore, pairwise image comparisons are required with the specific implementation depending on the used place representation (see section 3.3 for details). In case loop-closures are detected, links are added to the graph, and the position estimation technique can update the position estimates. The approach by [472] incrementally builds a topo-metric map with a feature-based place representation with position information obtained from the global positioning system (GPS).

Trajectory-based SLAM techniques are closely related to visual odometry because both techniques reconstruct the robot's trajectory. The method by [172] is a signature-based method for visual odometry. It incrementally integrates estimates of the robot's motion between consecutive images to derive an estimate of the robot's current position, but —in contrast to trajectory-based SLAM methods— it does not involve posterior position corrections. For [172], the robot's change of orientation between consecutive images is estimated by a phase-based compass (section 3.4.2.2), and the difference between the Fourier signatures of the current and the previous image are related to an estimate of the distance between the positions.

Localization

Regarding localization in topo-metric maps, most methods are Monte-Carlo methods for *incremental localization* which track the robot's position over time. For this purpose, the robot's state is tracked by fusing odometry information and visual information by a particle-filter framework (section 3.3.2.1). Each particle represents a possible robot pose, and the particle's importance is determined depending on how similar the currently perceived camera image is to the images stored in neighboring place nodes. Thus, efficient image comparisons are an essential aspect of localization with topo-metric maps. The importance factor is used to resample (more important particles have a higher probability for reproduction) and to derive an estimate of the robot's position (usually computed by weighted averaging). Signature-based approaches operating on sparse and dense topo-metric maps include [295, 427] and [171, 271, 377, 426, 505, 506, 538], respectively. The feature-based approaches [7, 8, 198, 619] localize the robot in sparse topo-metric maps. All these methods allow for continuous localization. Only the method by [200, 724] relies on discrete localization and applies a grid-based estimation technique (section 3.3.2.1) for keeping track of the robot's discrete state. In contrast to these *incremental localization* techniques, approaches for *global localization* estimate the robot's position directly from a single camera image without prior position information. For this purpose, the method by [421] relies on a signature-based place representation, whereas the methods by [474, 476, 477, 561] rely on a feature-based representation. The method by [68] is the only method which solves the localization problem by geometrical considerations: it determines the most similar views in the robot's surrounding and estimates the current position based on the positions of these views and the bearing towards them.

Planning and Following Routes

To plan and follow routes, standard methods for trajectory-control can be applied (review: [462]; textbooks: [41, 586]). These methods use the robot's current position estimate to minimize the deviation from the desired trajectory. Thus, routes are not restricted to place nodes. This is in contrast to route-following in topological maps (section 3.6.4.1): there, routes are planned by standard graph-search algorithms (textbook: [118]) and are segmented into a set of intermediate goal positions. The map is built from the trajectory-based SLAM method by [441, 698] and is used for a delivery task in a dynamic office environment. To fulfill the task, the robot plans the route and follows it in order to approach the goal position. The method by [720] is the only method we are aware of which applies topo-metric route maps (in this particular case with a holistic place representation). Snapshots along the route are sampled at equally-spaced distances, and route following is achieved by aligning the robot's orientation with the stored reference snapshots. While the robot navigates along the route, the robot's traveled distance along the route is tracked. Since the robot's orientation and its distance along the route are estimated, the method relies on partial pose estimation. Route following by aligning the robot's orientation according to a local reference direction is closely related to route following based on *holistic spatial representations* (section 3.6.4.2). However, holistic spatial representations neither segment the route into intermediate snapshots nor do they incorporate position information.

Navigation in a Do-It-Yourself Store

The series of papers [255–258, 327, 338–340, 581] describes a robot assistant for a do-it-yourself store. The robot relies on omnidirectional visual information as primary sensory information (but also on range information) and on a topo-metric map of its environment. It uses a rotation-dependent signature built from color statistics (section 3.3.1.2) for purely vision-based Monte-Carlo localization [255, 256, 258], for Monte-Carlo localization fusing vision and range information [581], and for trajectory-based SLAM [338, 340]. The trajectory-based SLAM method by [338] does not estimate the robot's pose from a single image, but from matching a short series of images, i.e. a small local map, against the global map of the environment. This technique can increase localization accuracy in the store's repetitive and dynamic environment. Using a short image sequence was not only evaluated with signature-based methods but also for a feature-based representation of places [327, 339].

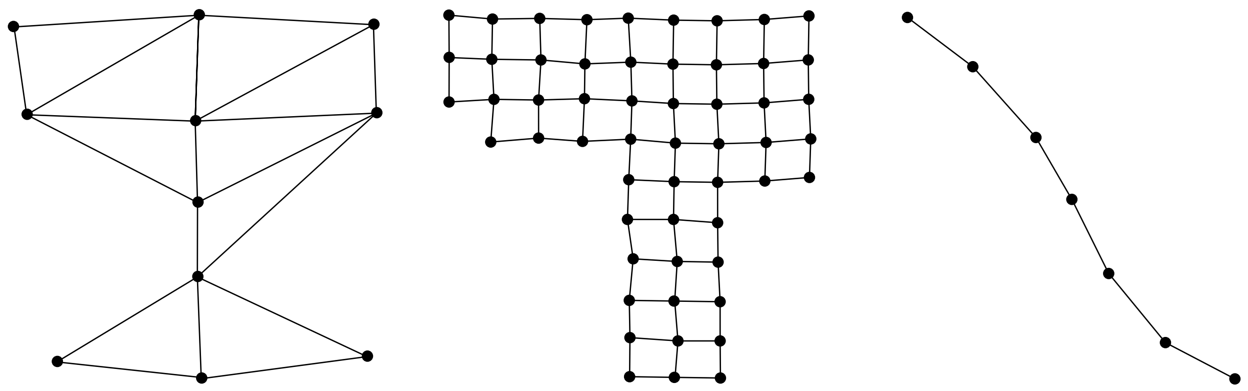
Cognitive Modeling
For cognitive modeling, a subdomain of biomimetic navigation, [424] proposes a robotic implementation of the spatial semantic hierarchy (SHH, reviews: [346, 347]). The method builds a hierarchical map inspired by the theory of cognitive mapping in humans. On the local level, a topo-metric map with feature-based place representation is learned which characterizes places by the corners resulting from the intersection of the robot's floor plane with walls or obstacles. For the global level of the hierarchy, the information of the topo-metric map is fused into a sparse model-based map (section 3.6.3.1) with a single frame, and the features' positions are given w.r.t. this world coordinate system.

Discussion of Topo-Metric Maps
Topo-metric maps are best used if the robot's task requires both accurate navigation and a sparse representation of space which scales well with the size of the mapped area. If required, maps can be built online and in real time. In contrast to model-based maps (section 3.6.3.1), topo-metric maps are an appearance-based representation of space and only allow for visualizing the positions of place nodes but not the position of visible objects (figure 3.35). Visualizations of the robot's environment would require additional computational effort to convert the topo-metric map into a model-based map —e.g. by identifying visual features and estimating their position in space by triangulation (section 3.3.2.1). In case the geometry of the robot's environment is known (e.g. from a floor plan), the topo-metric map can be visualized together with the environment if the rotation and the translation between the map's and the environment's coordinate systems are known.

In our opinion, topo-metric maps are the type of map suited best for low-level navigation of an autonomous floor-cleaning robot. In order to keep the robot at a fixed distance to the previous lane, some sort of metrical position information is required. In case of the trajectory controller described in chapter 4, this is a partial pose estimate which includes the robot's distance to the previous lane and its current orientation w.r.t. world coordinates. In later work, a full pose estimate is computed by Kalman filtering [314] or particle filtering [457]. Because we apply a warping method (section 3.5.2.2) to estimate spatial relations, the used maps belong to the class of *topo-metric maps with holistic place representation.* At the current state of our work, our methods do not involve subsequent corrections of position estimates inherent to SLAM methods. Our navigation methods are sufficiently accurate to build locally consistent maps. Complete coverage of complex shaped workspaces can then be obtained by combining several locally consistent sub-maps to a hierarchical representation of space. Thus, it avoids the computational effort of subsequent position corrections (section 7.3.3). For the low-level aspects of cleaning-robot control considered in this thesis, it is sufficient to visualize the topo-metric map (and not the robot's environment). A visualization of the environment could nevertheless be relevant for an advanced user interface, e.g. for guiding the robot to a position in space by marking the position in a map of the robot's environment.

3.6.4. Literature Review on Qualitative Maps

In contrast to quantitative maps, qualitative maps model the robot's workspace without position information and directly operate on the raw sensor data or information derived directly from it (in contrast to model-based maps operating on the spatial positions of visible image features; section 3.6.3.1). Thus, such navigation methods lack the steps of fusing sensor data into a common frame of reference and of inferring a metrical estimate of the robot's position as they are required by



(1) Sparse topological map. This is the most common type of topological maps.

(2) Dense topological map. Such maps sample the robot's environment more densely and often more regularly than sparse topological maps.

(3) Topological route map. Route maps are topological maps which are not branched.

Figure 3.36.: Types of topological maps. Subfigures (1) to (3) depict sparse topological maps, dense topological maps, and topological route maps, respectively. Due to the lack of metrical position information, only the topology of places is known. Thus, only their spatial interrelations but not their exact spatial positions can be visualized. The topologies of the maps visualized in subfigures (1) to (3) are identical to the topologies of their topo-metric counterparts depicted in figures 3.35.1 to 3.35.3.

quantitative navigation methods (section 3.6.3). For this reason, qualitative methods are in many cases computationally less demanding than quantitative methods [175, 189, 432].

Qualitative maps can be divided into *purely topological maps* (section 3.6.4.1) and *holistic spatial representations* (section 3.6.4.2). The former are a graph-based representation of space with nodes representing places and links expressing direct reachability between the adjacent places (figure 3.36). We decided for the term *purely topological maps* to emphasize the difference to *topo-metric maps*. Navigation methods relying on *holistic spatial representations* learn direct associations between the robot's current perception and action. Thus, in contrast to purely topological maps, the current sensor data is associated with an action required e.g. for accomplishing a route-following task and not with a position in space. *Holistic spatial representations* must not be confused with *holistic place representation* (section 3.3.1.1). In the following sections 3.6.4.1 and 3.6.4.2, we will further describe purely topological maps and holistic spatial representations.

3.6.4.1. Purely Topological Maps

Like topo-metric maps (section 3.6.3.2), purely topological maps are usually sparse, graph-based representations of space using former robot positions as landmarks. The nodes of the graph represent known positions in space characterized by image information, and two nodes are linked if the corresponding places are directly reachable from each other. As purely topological maps do not contain an estimate of the robot's position, it is not possible to distinguish different places with identical visual appearance as it is the case for topo-metric maps (section 3.6.3). This makes purely topological maps more prone to perceptual aliasing than quantitative methods. To circumvent this drawback, a short history of previously visited places can be used to disambiguate different places with identical visual appearance. Furthermore, only the topology but not the geometrically correct spatial arrangement of the map can be visualized (figure 3.36). Geometrically correct visualizations of purely topological maps usually rely (i) on external position information (e.g. obtained from an external robot tracking system; figures 4.15 and 6.15) or (ii) on reintroduced metrical information yielding a topo-metric map (section 3.6.3.2).

Like topo-metric maps (section 3.6.3.2), purely topological maps can be subdivided according to the graph structure and depending on the method used to characterize places by the available visual information. Regarding the graph structure, one can distinguish *sparse topological maps*, *dense topological maps*, and *route maps*. The most commonly used type of maps are sparse topological maps (figure 3.36.1). In case the application cannot be accomplished with sparse maps, e.g. if the task requires precise navigation, dense topological maps can be applied [313, 421] sampling the workspace more densely and often grid-like (figure 3.36.2). In case only a single route should be represented, a route map is sufficient to accomplish the task (figure 3.36.3). It segments the route into a list of intermediate goal positions with intermediate positions being only linked to their direct predecessor and successor. Thus, it can be understood as topological map without branches. In case the route is blocked, the robot cannot find an alternative path to the goal [189]. Depending on how visual information is used to characterize places (sections 3.3.1.1 to 3.3.1.3), purely topological maps can be subdivided into purely topological maps with *holistic*, *signature-based*, and *feature-based* representations of places. Like for topo-metric maps (section 3.6.3.2), we do not consider these groups separately to avoid repeated descriptions. We rather focus on the applications of purely topological maps, namely map building, localization, route following, and biomimetic navigation.

Map-Building

Purely topological maps can be built by *offline methods*, *online methods*, or *topological SLAM methods*. For all three categories, accurately detecting loop-closures is essential for consistent mapping. Loop closures are detected by comparing the currently perceived image with the images stored in the map. This step strongly depends on the used type of place representation; for details on techniques for recognizing places please refer to section 3.3. For *offline mapping* methods, the robot first explores its workspace and collects image data without building a map. In a second step taking place after the robot completely explored its workspace, the map is built by determining place nodes and linking neighboring nodes based on the visual information gathered in the first step. The drawback of offline methods is that the robot cannot use the map while exploring its environment. The advantage is that maps resulting from offline methods are often considered to be more accurate or to better represent the robot's environment because mapping (i) only takes place if the entire visual information is available and (ii) is not subject to real-time constraints. The latter approach allows to apply more complex map-building algorithms. The methods by [54–56, 58, 343, 725, 726] rely on a feature-based representation of places; the algorithm by [481] relies on a holistic place representation. Among these methods, [54–56, 343, 725, 726] describe an approach to hierarchical mapping (section 3.6.2). The algorithm builds a sparser high-level representation from grouping similar images at the lower map level (figure 3.32.3).

Online methods map the environment while the robot explores its workspace. To accomplish this task, the robot needs to concurrently perform localization and map-building. In simpler cases algorithms for online mapping add links and or place nodes following deterministic rules and do not remove links and place nodes after adding them to the map. Examples include [666] relying on a holistic place representation, [210, 211, 380, 473] relying on signature-based place representations, and [237, 555, 651–653] relying on feature-based representations of places. *Topological SLAM algorithms* do not determine the topology based on deterministic rules but by applying probabilistic techniques. While mapping, they repeatedly estimate the most likely graph topology given the currently available sensor by means of statistical inference. Topological SLAM algorithms use signature-based place representations [381, 530, 621–623, 686, 687] or feature-based place representations [126–128, 488, 531] in combination with an efficient image matching methods (section 3.3.1.3). The method by [127, 128] was shown to reliably estimate the map for trajectories up to 1000 km length.

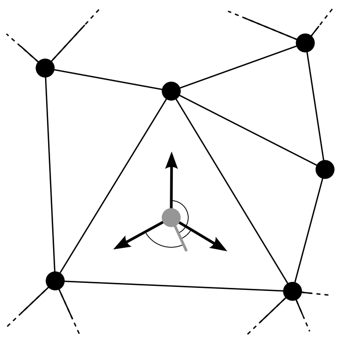


Figure 3.37.: Approaching arbitrary positions in topological maps. An arbitrary position (gray filled circle) can be characterize by the bearing (black arrows) towards three surrounding place nodes (black filled circles) of the topological map. The place nodes are used as landmarks, and bearing information can be computed by local visual homing and uniquely defines the position. To the best of our knowledge, we are not aware of a navigation strategy exploiting this principle with purely topological maps. Please note the similarity to the principles underlying local visual homing shown in figure 3.22.

Localization

Localization in purely topological maps means to compare the currently perceived image with the reference images stored in the map and to retrieve the best match. The best match is determined by place recognition strategies reviewed in section 3.3. Each place node of the map is surrounded by a certain region, in which it is determined as best matching place node. Within such a region, two images acquired at different spatial positions cannot be distinguished by qualitative localization methods. This property distinguishes qualitative localization from quantitative localization which computes a continuous position estimate. The only possibility to achieve a finer spatial resolution is to sample snapshots more densely. This strategy leads to dense topological maps (figure 3.36.2).

In contrast to the field of topo-metric localization (section 3.6.3.2) where many approaches perform incremental Monte-Carlo localization (i.e. the estimate of the robot's position is iteratively updates), the approaches to purely topological localization are global techniques (i.e. they determine the most similar place node without tracking the robot's position over time). Hence, the position estimate is derived from matching a single camera image without prior knowledge. Since global localization requires to compare a large number of image comparisons, most localization methods rely on signature-based place representations (sparse maps: [94, 169, 242, 311, 312, 352, 500, 541, 647, 688], dense maps: [313, 421]). Methods relying on feature-based place recognition include [18, 199, 529, 555, 651, 655]. For localization relying on a holistic place representation, a minimalistic method is proposed by [440]. It localizes the robot along a known route by matching a short image sequence against the images stored along the route. This allows for robust and accurate localization with low-resolution images: the maximum resolution tested in the paper is a total of 512 pixels.

Planning and Following of Routes

Methods operating on purely topological maps cannot rely on standard approaches for trajectory control (review: [462]; textbook: [41, 586]) for path following or approaching positions in space because they do not incorporate position information. In principle, methods operating on purely topological maps are theoretically capable of approaching arbitrary positions by using surrounding place nodes as landmarks (figure 3.37). Although this principle is commonly applied by strategies using topo-metric maps, we are not aware of a method exploiting this principle with purely topological maps. The methods we are aware of restrict navigation to the places stored in the map, and route following is achieved by moving along a sequence of intermediate snapshots (figure 3.36.3). Such sequences (or routes) are either learned from a teaching process or selected by a path planning algorithm (e.g. Dijkstra's shortest path algorithm; textbook: [118]) operating on the purely topological map. In the former case [14, 136–138, 235, 236, 355, 658], the robot is only capable of following the route taught e.g. by manually guiding the robot along the route. The latter case [210, 211, 237, 379] is much more flexible and allows for navigating between arbitrary points of the map. In both cases, the robot navigates from intermediate position to intermediate position by applying local navigation strategies (usually local visual homing methods; section 3.5) until the goal is reached. Route following methods include [210, 211, 355, 658, 704–706] operating on a holistic representation of places, [136–138, 592–594] operating on signature-based representations, and [121, 235–237, 379, 620, 659] operating on a feature-based place representation. The method by [704–706] implements route following by aligning the robot with the stored reference images. Due to this, the method is closely related to the topo-metric route following algorithm by [720] (section 3.6.3.2) and the method proposed by [21–24] relying on a holistic spatial representation of space (section 3.6.4.2). The methods by [210, 211, 235–237] both rely on a hierarchical representation of space using a purely topological map as high-level representation. While moving between two consecutive place nodes along the route, a sparse model-based map is built (section 3.6.3.1).

Biomimetic Navigation
In the context of biomimetic navigation, the method by [266, 268, 269] models the processes involved in the rat's hippocampus during navigation by learning a purely topological map with a holistic representation of places. The map is learned by vector quantization of camera images (in this particular case by applying a self-organizing map; textbooks: [46, 148]), and links are added depending on the image similarity of the linked views. The map can be converted into a topo-metric map (section 3.6.3.2) by reintroducing distance information followed by map relaxation [266, 269]. The series of papers [228–231] describes a hippocampus model operating on a feature-based place representation. It learns a set of place nodes representing a region in the robot's environment (in terms of hippocampus models termed place cells and place fields, respectively), and associates each place node with an action required to fulfill the robot's task. The model is applied to route following and approaching a goal position. The latter task is closely related to local visual homing (section 3.5.1), but the goal position is not defined by a single snapshot but rather by a set of snapshots acquired surround the goal position.

Discussion
Purely topological maps are best used for tasks which only require coarse navigation or which can be solved without position information. The main advantage of purely topological maps is that both building and operating on such maps requires little computational effort. However, visualization is difficult because visualizations usually rely on external position information or on reintroducing metrical position information. Approaching places is restricted to places already stored in the map (although at least theoretically arbitrary places could be approached; figure 3.37). For navigation of an autonomous cleaning robot, purely topological maps are not applicable because they do not contain metrical position information. Metrical position information is required for keeping the robot at a predefined distance to its previous lane.

3.6.4.2. Holistic Spatial Representations

Holistic spatial representations are strongly influenced by recent work on modeling route-navigation capabilities of desert ants ([21, 245, 703] and sections 3.3.3.3, 3.4.3.2 and 3.5.3.1). The term *holistic spatial representation* follows the description *holistic (route) representation* originally used by [21–23, 512]. It refers to a spatial representation aggregating information from several places of the robot's environment and must not be confused with *holistic place representations* (definition: section 3.3.1.1; examples: sections 3.6.3.2 and 3.6.4.1) or *holistic local visual homing methods* (section 3.5.2.2).

The key idea of holistic spatial representations is to directly link the current sensory information to an action and not to a position such as a graph node for topological maps or a metrical position estimate for quantitative maps. Because of the direct coupling between sensor data and an action, navigation strategies operating on holistic spatial representations are based on *recognition-triggered responses* (review: [189, 403, 642]). The knowledge required to accomplish the robot's task is integrated into a holistic representation storing perception-action links in an aggregated way without relating them to a position in space. Thus, the resulting representation does not segment the

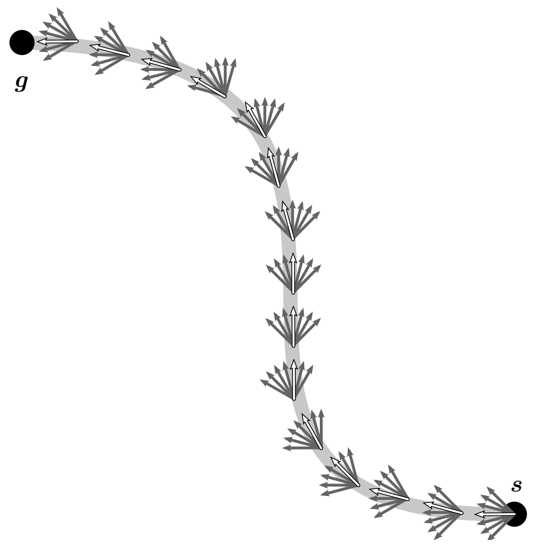


Figure 3.38.: Sketch of the route-following method by [21–23, 512] based on a holistic spatial representation of space. The robot follows the learned route (thick light-gray line) from the start position ***s*** to the goal ***g***. Therefore, it pursues scanning movements either by physically rotating the robot or by mentally rotating the perceived image. For each tested orientation (gray arrows), the familiarity is determined by consulting the holistic spatial representation. By moving into the most familiar direction (open arrow) and by repeating this procedure, the robot follows the route until it reaches the goal position. Because of the abstract nature of the holistic spatial representation, we cannot visualize the representation itself but only the action linked to the visual information perceived at a certain position in space. As the method does not compute the spatial positions of visible features, it is not possible to reconstruct the robot's environment.

environment into a set of known positions (as is e.g. the case for graph-based representations), and it cannot be subdivided into local submaps (e.g. for obtaining a hierarchical representation of the environment section 3.6.2). Since the perceptual information is not related to a position in space, holistic spatial representations cannot be applied for localization tasks.

Navigation strategies operating on holistic spatial representations require a learning phase to establish links between perception and action. Although the built representation differs from traditional maps, the learning stage is comparable to map building. During this learning stage, all the perceived sensor information is integrated into the map. Once the representation is built, the agent is capable of navigating over known terrain (e.g. following a learned route), but it is not capable if using the map for navigating over unknown terrain. While using the representation for navigation, it can be continuously updated by integrating new visual information. This property could in robotic applications e.g. be used for adaptive approaches to increase robustness against illumination or scene changes (section 3.2.3.2). Although representing the robot's environment more abstractly than traditional approaches, we consider holistic spatial representations to be appearance-based navigation methods. Holistic spatial representations were proposed as biologically plausible models of route-following in insects and are compact representations of space and allowing for developing parsimonious navigation strategies (section 3.6.5.2).

For the route-following strategy proposed by [21–24] (review: [512]), the holistic spatial representation is learned and represented by an artificial neural network providing a familiarity measure for the currently perceived view (different neural networks and image representations are tested in the different papers). Independent of the which representation is used, the route-following relies on the following scheme (also depicted in figure 3.38): while moving along the learned route (figure 3.38, thick light-gray line), the agent (in this case an animal or a robot) performs rotational scanning movements (gray arrows), determines the familiarity of these scanning view using its holistic spatial representation, and moves into the direction of the most familiar view (white arrow). By repeating this procedure, it is guided along the route towards the goal position. The method was demonstrated to allow for reliable and robust route-following in simulations [21, 23] and real-robot experiments [22, 24]. Compared to the other reviewed navigation methods, the route following methods by [666, 720] are the most similar to the route-following method based on a spatial holistic representation. These methods achieve route following by orientation aligning with a reference direction [720] and by a tight coupling between perception and action [666]. However, both methods segment the route into a series of intermediate goal positions; the method by [720] also relies on metrical position information.

Holistic spatial representations are strongly influenced by research on route following in insects and reflect recent findings in this field. As the real-robot experiments by [22, 24] were conducted in an environment differing strongly from office environments, apartments, or outdoor environments, predicting the method's performance for such environments is difficult. Holistic spatial representations are limited to navigation along a previously taught route, and it is currently unclear if they can be extended to a more flexible representation storing several routes and allowing to flexibly select routes and paths. We expect that navigation strategies operating on holistic spatial representations do not achieve the accuracy and repeatability of quantitative navigation methods (section 3.6.3). Where suitable for the robot's task, holistic spatial representations offer a parsimonious navigation methods with respect to both computational power and memory requirements. Holistic spatial representations are not suited for navigation of cleaning robots because (i) they only allow for navigation over known terrain and (ii) they do not contain metrical position information. Because of the latter aspect, they do not allow to keep the robot at a constant distance to its previous lane, which is an essential prerequisite for cleaning-robot control.

3.6.5. Discussion of Map-Based Navigation

In the following, we will discuss (i) general aspects of map-based navigation (section 3.6.5.1), (ii) the relation between map-based navigation in robotics and research in the field of spatial cognition (section 3.6.5.2), and (iii) the application of map-based navigation in this work (section 3.6.5.3).

3.6.5.1. General Aspects of Map-Based Navigation

In most cases, the choice of a specific type of map is already determined by the used representation of places (section 3.3). The different possibilities to represent places based on visual navigation as summarized in figure 3.16 exactly fit the categorization of mapping methods depicted in figure 3.30 —except for holistic spatial representations (section 3.6.4.2). These are a special case because they do not explicitly represent places but directly link the perceived visual information to an action.

Former reviews [175, 189, 432] and textbooks [586, 630] describe a clear separation between *quantitative* and *qualitative* navigation. According to these, *quantiative* methods are computationally more complex but also allow for more accurate navigation, whereas *qualitative* methods are computationally less demanding and typically applied for coarse, i.e. less precise, navigation. We are of the opinion that this strict separation has softened during the period of time considered for this chapter. On the one side, there are *quantitative* methods operating in real-time and —on the other side— there are *qualitative* methods involving complex image-processing or map-estimation steps. We also feel that topo-metric maps (section 3.6.3.2) and hierarchical maps (section 3.6.2) contributed to reducing the gaps between these two types of maps. The effect could also be due to the progress in computing power made during the last ten years, particularly in the field of mobile or embedded processors which now allow for complex image-processing operations in real-time (e.g. [17, 357, 513, 673, 674]).

Beyond that, substantial achievements were obtained with respect to the size of the mapped environment. While early work was restricted to single rooms or smaller office environments, recent methods are capable of mapping large-scale environments such as entire apartments, large office environments, or outdoor environments like entire cities. Such large scale maps can be built from thousands of snapshots collected along trajectories up to 1 000 km in length [127, 128].

3.6.5.2. Map-Based Navigation and Spatial Cognition

Animals and humans also rely on map-based navigation strategies to achieve long-range navigation. We outline the current opinion regarding these navigation capabilities of flying insects (mostly bees), ground-living insects (ants), rodents, and humans. Despite their tiny brains with approximately

100 000 (fruit fly: [104]) to 1 000 000 neurons (honey bee: [428]; but please note the comment in [105] on comparing organisms by the number of neurons) and their low-resolution visual system, insects show amazing navigation capabilities. To give just one example among many others, honey bees were observed to successfully return to their hive after being displaced for up to 13 km [497]. These are exactly the capabilities which have drawn the attention of engineers to mimic design principles of insects to implement parsimonious but robust visual control algorithms (recent reviews: [250, 596, 597]). Honey bees can rely on different senses to obtain the environmental information required for navigation among which vision is of special importance (review: [596], textbook: [625]). It is subject of current research to investigate the influence of different visual cues (e.g. color, patterns, or entire panoramas; [596, 717]) and determining which cues are used in certain situations. The same holds for the underlying spatial representation used to memorize the information required for navigating: the authors of [429–431, 480] argue that shortcutting behavior, i.e. the ability to find unknown shortcuts between known places, requires a cognitive map. Cognitive maps are maps with a common frame of reference and are therefore similar to *quantitative maps* (section 3.6.3). However, these results cannot be seen as a proof for such a map because (i) the modeling study [125] reveals that identical behavior can be obtained without a common frame of reference, (ii) optimal path planning of bumblebees between feeders including shortcutting can also be explained without such a map [373], and (iii) it cannot be ruled out that the results are not due to the bees using distant visual cues for navigation (comment by [I42] on [431]).

For central-place foragers such as bees and ants, route-following behavior is essential because these animals shuttle back and forth between the nest and a food source, carrying food to the nest. For ants, several studies (e.g. [249, 336]) exist describing how ants learn routes. In contrast to bees it is accepted that they do not rely on a cognitive map [102, 115, 681]. Whereas earlier work (e.g. [93, 315, 598]) explained route navigation by following a sequence of snapshots (comparable to a *route map*; sections 3.6.3.2 and 3.6.4.1), recent findings [245, 511, 512] suggest a *holistic spatial representation* (section 3.6.4.2) of the ants' environment without explicitly storing images. Assuming that the holistic spatial representation can be used to determine the familiarity of the currently perceived view, route following can be explained by performing rotational scanning movements and moving into the direction of the most familiar view. Although modeling studies reveal that the observed behavior can be well reproduced [21–23, 512], it is not yet proven that ants rely on this concept for route following. Further recent studies investigating route-following capabilities of ants include [536, 699, 700].

Regarding rodents, most of the work is neurophysiological research investigating the role of the hippocampal place-cell system for navigation. The interplay of three different types of neurons, namely place cells, grid cells, and head-direction cells, is assumed to be essential for forming a spatial representation of the environment. Nevertheless, many details still need to be investigated; for an overview of the current state of research in this domain, the reader is referred to the reviews [4, 279, 465]. The methods by [266, 268, 269] and [228–231] rely on a computational place-cell models for long-range navigation using omnidirectional vision as primary sensory information (see e.g. [28, 29, 78] for other sensor modalities). The former [266, 268, 269] are more abstract models relying on a holistic place representation together with a *purely topological map* (section 3.6.3.2) or —after reintroducing metrical information— with a *topo-metric map* (section 3.6.4.1). The latter method [228–230] applies a more detailed model learning a purely topological map with feature-based place representation.

For humans, the situation is even more unclear. This is probably due to (i) the large amount of different navigational tasks (see [690] for a taxonomy of tasks) humans can perform and (ii) the many different visual cues humans can rely on for navigation. These two aspects make it difficult to experimentally prove or validate hypotheses. Different tasks could even be solved using different spatial representations. According to [403], it is likely (i) that humans rely on a graph-like and hierarchical spatial representation with local metrical information and (ii) that humans do not integrate their spatial knowledge into a map with a single global frame of reference.

3.6.5.3. Application of Map-Based Navigation in This Work

A cleaning robot needs a map of its environment to remember cleaned areas, to detect and approach uncleaned areas, and to efficiently return to its charging station (chapter 2). Furthermore, it requires some sort of metrical position information in order to cover its workspace by meandering and parallel lanes which are a predefined distance apart from each other. The latter prerequisite excludes the application of *qualitative* maps for our particular task (section 3.6.4). Among quantitative maps, we consider *dense model-based maps* to be not appropriate for our task because these maps are mainly used for visualization purposes and require a computational power, which we think is not available on a mobile floor-cleaning robot (section 3.6.3.1). *Sparse model-based maps* are suited for navigation of domestic floor-cleaning robot and are applied by related work (section 2.2.2.2) and probably also by some commercially available robots (section 2.2.1). However, this type of map requires (i) a feature-based representation of places and (ii) the application of position estimation techniques to compute and maintain the positions of visible features (sections 3.3.1.3 and 3.3.2.1). Both techniques are usually computationally demanding even though many efficient techniques were proposed during the last years. Beyond that, additional effort is required to memorize the already cleaned area (e.g. by storing the robot's trajectory or a polygon representing the already cleaned area). Among all types of maps, we consider dense topo-metric maps (section 3.6.3.2) to be best suited for our purposes. Such maps can be easily built from the available visual information, allow to apply our 2D warping method for local visual homing (section 3.5.3.2), offer a straight-forward representation of already cleaned areas, and allow for efficiently detecting already cleaned areas (chapters 5 and 6).

3.7. Overall Discussion and Outlook

The progress is visible for any level of the proposed hierarchy of navigation strategies, but it is most prominent for state-of-the-art methods which allow for large-scale navigation under real-world conditions (e.g. [127, 339, 568, 655]). During the period of time considered for this review, i.e. from the year 2000 until April 2013, substantial progress was made in the field of navigation of wheeled mobile robots using omnidirectional vision as primary sensor information. We see the main driving forces of these achievements in the following three aspects: (i) the advances made in the field of local image features including their detection, description, and efficient matching methods (section 3.3.1.3), (ii) the application and improvement of position-estimation frameworks (section 3.3.2.1), and (iii) the tremendous increase in computing power which makes real-time application of these techniques possible. Besides the improvements of navigation capabilities, omnidirectional vision setups have been established as sensors for wheeled mobile robots and as an interesting alternative to both traditional directed cameras and range finders. Due to the vast progress in camera technologies, recent cameras are tiny and low-budget sensors yet offering an amazing image quality. We expect that this miniaturization of cameras will allow to construct more compact omnidirectional vision setups.

To further advance the field of mobile robot navigation relying on omnidirectional vision, we currently see three main working directions for future work: (i) the comparability of navigation strategies (section 3.7.1), (ii) the applicability of strategies under real-world conditions (section 3.7.2), and (iii) the application of omnidirectional visual navigation strategies to solve real-world applications (section 3.7.3).

Table 3.3.: Available omnidirectional image databases. The given references point to download links and to papers describing the corresponding database.

Database	Download	Description
COsy Localization Database (COSY)	[I78]	[522]
From Sensors to Human Spatial Concepts (FS2HSC)	[I113]	[343, 725, 726]
Rawseeds repository	[I79]	[48, 89, 415]
SIFT, SURF, and Seasons	—[1]	[651, 652, 654, 655]
Oxford dataset	[I98]	[126, 488, 595]
Bielefeld databases	[I15]	[342, 454, 660]
OpenSLAM repository[2]	[I91]	—
Radish repository[2]	[I45]	—
Mobile Robot Programming Toolkit (MRPT)[2]	[I48]	—

[1] Available upon request.

[2] Mainly contains laser-range data, but can also provide omnidirectional image data.

3.7.1. Comparability of Navigation Strategies

A good comparability of navigation strategies is important to identify and compare their particular strengths and weaknesses. We currently see two approaches to achieve a good comparability, namely *sharing image databases* (section 3.7.1.1) and *benchmarking methods* (section 3.7.1.2).

3.7.1.1. Sharing Image Databases

Image databases are collections of (omnidirectional) images collected by a real-robot taken in a real environment. Since the images are collected with real camera hardware, they allow for performing more realistic simulation experiments than would be possible for simulations relying on virtual-reality techniques. The robot's positions and orientations when images were acquired are usually known (e.g. by relying on an external tracking system; figures 4.15 and 6.15) and can be used as ground truth for data evaluation. The image databases allow for conducting experiments under identical experimental conditions, which is important for comparing different navigation strategies (or for parameter optimization). Several research groups already make their image databases publicly available (table 3.3) or reuse existing databases —a good practice which should be continued in the future. Beyond that, new dataset should be collected which cover situations, such as abrupt illumination changes or strong dynamic changes of the scene, for which current navigation strategies fail. These datasets can then be used as a starting point to improve existing navigation methods.

In the context of this thesis, existing image databases containing images collected along a regular grid ([I15] and [342, 454, 660]) are used for visual detection of already cleaned areas (chapters 5 and 6). The dataset of unfolded and rotated images from these databases used for our experiments can be downloaded at [S1]. Furthermore, Claudius Strub in cooperation with Lorenz Hillen and Martin Krzykawski developed a method for closed loop control of a robot by integrating external position information from an active robot-tracking system (figure 6.15) into the particle-filter framework by [457]. This method will in future work be used to collect high-precision image databases under operating conditions which are currently challenging to solve for existing navigation strategies.

3.7.1.2. Benchmarking Methods

Benchmarking methods allow for a sound statistical evaluation of experimental results obtained from navigation strategies and therefore improve the comparability of results. Existing benchmarking methods (e.g. [27, 349, 397, 697]) were developed for comparing maps based on range data, but we expect that similar approaches can be used to compare maps built from omnidirectional visual information. We hope that more and more groups will evaluate their data with such thorough benchmarking methods rather than presenting only a simple demonstration of functionality. Throughout this thesis, we have spent considerable effort to derive and implement quantitative performance measures for all our database (chapters 5 and 6) and real-robot experiments (chapters 4 and 6).

3.7.2. Applicability Under Real-World Conditions

Real-world operating conditions include changes of the illumination and dynamic changes of the scene (section 3.2.3.2). If at all, existing methods only cope with one of these issues (illumination changes: [8, 136, 138, 506, 538, 651, 652, 654, 655]; dynamic scene changes: [133–135, 282, 297, 306, 379, 422, 423, 441, 698]). Reliable and robust navigation under real-world conditions not only requires capabilities to cope with all of these aspects together and not just with a single one. From today's perspective, there are still a lot of challenges to solve before most navigation strategies relying on omnidirectional visual navigation can completely deal with real-world operating conditions. For the work presented in this thesis, we concentrate on achieving robustness against illumination changes (chapters 5 and 6) and assume static environments —although this assumption is only partially valid for navigation of domestic floor-cleaning robots. Methods for achieving robustness against dynamic scene changes are left for future work (section 7.3.1).

3.7.3. Towards Real-World Applications

Solving complex real-world applications such as autonomous floor cleaning or guiding tourists in a museum requires not only a combination of navigation capabilities (e.g. mapping, map visualization, localization, and path planning) but also needs to include further aspects such as usability, user interaction and safety. Currently, the vast majority of papers describe only a single building block of such a complex system. Only few groups work on more complex robot systems capable of solving real-world applications relying on omnidirectional visual information. Such applications include a shopping assistant for do-it-yourself stores ([255–258, 327, 338–340, 581] and section 3.6.3.2), an autonomous wheel chair (still operating in a lab environment; [235–237] and section 3.6.4.1), or our project on navigation of autonomous cleaning robots (e.g. [214, 215, 222, 223, 453, 457]). We think that the surprisingly small number projects dealing with real-world applications or parts thereof is a hint for the complexity and difficulty of such research projects. Nevertheless, from our experience on navigation strategies of autonomous cleaning robots, we conclude that focusing on a concrete real-world application and developing reliable and robust navigation strategies to solve the application —instead of solely tackling partial aspects independent from a concrete real-world application— can tremendously contribute to the progress in the research domain.

We consider the past 15 years of research on omnidirectional vision as essential for (i) developing omnidirectional sensors, (ii) improving and establishing techniques such as feature-based approaches and algorithms for position estimation, and (iii) developing methods to solve navigation tasks such as mapping or localization. For the upcoming 15 years, we hope and expect that more and more robotic systems will be developed solving complex real-world scenarios using omnidirectional vision as primary sensory information.

4. Trajectory Controller Based on Partial Pose Estimation and Dense Topo-Metric Maps

In this chapter, we describe a mostly vision-based trajectory controller for guiding an autonomous floor-cleaning robot along meandering and parallel lanes. The controller relies on a dense topo-metric map with partial pose information.
Section 4.1 introduces the main ideas of the proposed navigation strategies. Dense topo-metric maps and the trajectory controller are described in sections 4.2 and 4.3, respectively. The experiments and the obtained results are described in sections 4.4 and 4.5. The results are discussed and conclusions are drawn in section 4.6, and section 4.7 points out future working directions.
The method was initially developed and tested in simulations during the course of the diploma theses of Dr. Sven Kreft [342] supervised by Frank Röben and Lorenz Hillen. On this basis, Frank Röben and Lorenz Hillen pursued an initial implementation on a Pioneer robot, which was then improved by Daniel Venjakob. Martin Krzykawski was responsible for porting the code to our custom-built cleaning robot. Lorenz Hillen conducted experiments, implemented the software for data evaluation, and prepared the publications [214, 222]. The visual tracking system (section 4.4.5 and figure 4.15) used to track the robot during the experiments was initially developed during a student project [159] by Johann Engelbrecht, Martin Höner, Andre Lemme, and Ioannis Moutogiorgos under the supervision of Frank Röben and Lorenz Hillen. The system was further improved by Frank Röben and Martin Krzykawski (low-level aspects of client-server architecture) and Lorenz Hillen (high-level aspects regarding tracking robustness and accuracy, usability, and data logging).
Except for minor modifications, this chapter is identical to sections 4.2 to 9 of our article GERSTMAYR-HILLEN *et al. [222] published in "Robotics and Autonomous Systems". Beyond that, the method was —without detailed mathematical description— presented at conferences (*GERSTMAYR *et al. [214] and* MÖLLER *et al. [453]) and a workshop (*GERSTMAYR, RÖBEN, *and* MÖLLER *[215]), described in the dissertation of Frank Röben [537], and patented [219].*

4.1. Introduction

This chapter describes a navigation strategy for an autonomous floor-cleaning robot mapping its workspace while moving along parallel and meandering cleaning lanes. The map is not only extended but also concurrently used by the proposed trajectory controller to obtain parallel lanes at a predefined distance. In the following, we will briefly outline the two essential aspects of this chapter, namely *map building* and *trajectory control* for autonomous cleaning robots (sections 4.1.1 and 4.1.2).

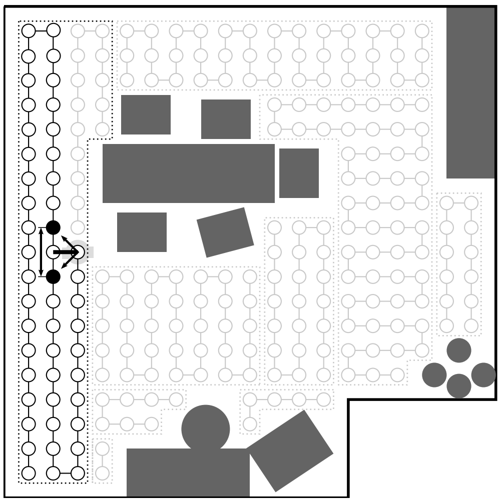


Figure 4.1.: Key ideas of the proposed trajectory controller. By taking the bearing to two snapshots stored along the previous lane (black filled circles) and by combining the angular information (arrows) with an estimate of the distance between the considered snapshots (double-tipped arrow), the robot's distance (thick black line) to the previous lane can be estimated. Complex-shaped workspaces can be incrementally covered completely by combining several cleaning segments (dotted areas) of parallel and meandering lanes as resulting from the control strategy proposed in this chapter.

4.1.1. Mapping for Cleaning Robots

The map has to store all the information about the robot's environment necessary for accomplishing the robot's task at the current or later points in time, and the type of map has to be suitable for the robot's application as well as for the robot's sensory equipment. In our case, the robot (section 4.4.3) is equipped with an omnidirectional vision system (section 3.2.4) as main source of information. Among all types of maps reviewed in sections 3.6.3 and 3.6.4, we decided for *dense topo-metric maps with holistic place representation* (section 3.6.3.2). We think that such maps are the best choice for our purpose because (i) building such maps from omnidirectional images is possible in real-time even with limited computational resources, (ii) they allow to store position information which is required to keep the robot at a pre-defined distance from the previous lane, (iii) the trajectory controller can use our existing local visual homing algorithms (section 3.5) for estimating spatial relations between place nodes, and (iv) their dense spatial resolution allows to frequently update the controller. At later stages of the cleaning process, the map will provide the spatial information required (i) to detect and to approach areas which still need to be cleaned and (ii) to visually detect areas which already have been cleaned (loop-closure problem; chapters 5 and 6) at a fine spatial resolution.

4.1.2. Introduction to the Proposed Trajectory Controller

The trajectory controller proposed in this chapter uses the map to keep the robot on parallel lanes while concurrently extending the map. By this means, a part of the robot's entire workspace is covered by meandering and parallel lanes (figure 4.1, black dotted area). At the beginning of a cleaning run, the robot moves straight forward and successively adds place nodes (circles) to its dense topo-metric map. The lane ends if an obstacle is encountered, if it approaches an already cleaned area, or if it extends beyond the previous lane. In case the current cleaning segment can be extended, the robot turns and a new lane is started; otherwise a new segment is planned. From the second lane on, the controller not only stores place nodes while moving, but also uses neighboring snapshots (filled circles) stored along the previous lane to estimate its current distance to the previous lane. By applying a local visual homing algorithm, the bearing and the compass information to at least two snapshots (arrows) are computed. Furthermore, the spatial distance between the considered snapshots (double-tipped arrow) is determined from the robot's wheel odometry. Both sorts of information are then combined in order to estimate the robot's distance to the previous lane (thick black line). This estimate is passed to a controller generating a movement command keeping the robot at a constant distance to the previous lane and hence on a course parallel to this lane.

To estimate the robot's distance to the previous lane, we do not determine the robot's full pose w.r.t. world coordinates as is the case for most related approaches (sections 2.2.2 and 3.6.3). Rather, we only compute the information which is sufficient and necessary to fulfill the task. In our case, this includes estimates of the robot's distance to the previous lane and of its orientation. With only one of these estimates lacking, the robot would not be able to fulfill the task. For example, in case the distance estimate was missing, the robot would not be able keep its distance to the previous lane constant resulting (i) in a course parallel to the previous lane but following it at an arbitrary distance and (ii) in inter-lane distances varying from lane to lane. We refer to our approach as partial pose estimation in order to distinguish it from standard approaches estimating the robot's full pose (section 3.6.3.2). The advantage of partial pose estimation over full pose estimation is that the former is computationally less demanding —which we think is an important aspect if one considers the limited computational power of an autonomous cleaning robot. Due to this aspect, we also do not correct the partial pose estimates after adding the corresponding place node to the dense topo-metric map. This aspect distinguishes our approach from related work on visual simultaneous localization and mapping (SLAM; sections 2.2.2 and 3.6.1.2). Since the key aspect of our method is to incrementally estimate the robot's current position based on prior robot positions, it can also be considered a visual odometry algorithm (reviews: [191, 563] and section 3.6.1.2).

By the proposed controller, a single segment of parallel and meandering lanes can be covered. The robot's entire workspace can be completely covered by combining several of these cleaning segments (figure 4.1, light-gray); the strategies required for this purpose are subject of future work aiming at developing a more complex control scheme for an autonomous floor-cleaning robot (section 7.3.2). Thus, the presented trajectory controller and mapping algorithm can be understood as the basis of a system making an autonomous floor-cleaning robot relying on omnidirectional vision capable of cleaning complex areas.

4.2. Dense Topo-Metric Map

For a robot using omnidirectional vision as primary sensory input and relying on local visual homing to determine the spatial arrangement of landmarks, it is straightforward to build a topological representation of space. In the context of graph-based maps (sections 3.6.3.2 and 3.6.4.1), landmarks are images acquired at former robot positions. As described in section 3.6.4.1, purely topological maps are sparse representations of space, which do not contain metrical position information. We are of the opinion that the spatial resolution of such a sparse topological map is not suitable for our application because a certain number of snapshots is required to (i) control the robot's distance to its previous cleaning lane and (ii) to determine areas which still need to be cleaned and to find loop closures on a relatively fine level. These problems can be circumvented by sampling the robot's workspace more densely, thereby increasing the number of place nodes stored in the map. Furthermore, our particular task requires some sort of distance information in order to keep the robot at a predefined distance to its previous lane. This information is also stored in the place nodes of the map, therefore leading to dense topo-metric maps.

In our application, the dense topo-metric map is used to guide a floor-cleaning robot along parallel and meandering lanes. While moving along a cleaning lane i, a new place node $\mathcal{P}^{(i,j)}$ is added to the map whenever the robot's distance to the previous place node $\mathcal{P}^{(i,j-1)}$ exceeds the inter-snapshot distance Δs. For this purpose, the distance traveled since adding $\mathcal{P}^{(i,j-1)}$ is measured based on the robot's odometry. By this means, a regular and grid-like representation of the robot's workspace is built. Each place node

$$\mathcal{P}^{(i,j)} = \left\{ \boldsymbol{S}^{(i,j)}, \hat{\mathbf{s}}_{\text{odo}}^{(i,j)}, \hat{\theta}_{\text{vis}}^{(i,j)} \right\} \tag{4.1}$$

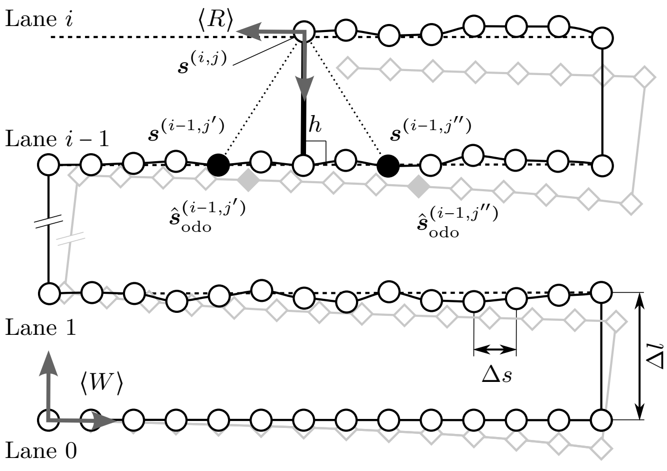


Figure 4.2.: Positions and coordinate systems involved in the proposed trajectory controller. The diagram shows the true robot trajectory (black line with circles), the trajectory based on odometry estimates (gray line with diamonds), and the ideal cleaning trajectory (black dashed line). Circles depict robot positions $\boldsymbol{s}^{(i,j)}$ at which snapshots $\boldsymbol{S}^{(i,j)}$ were taken; diamonds mark the corresponding odometry estimates $\hat{\boldsymbol{s}}_{\text{odo}}^{(i,j)}$. The robot's current distance h to the previous lane equals the height of the triangle defined by $\boldsymbol{s}^{(i,j)}$, $\boldsymbol{s}^{(i-1,j')}$, and $\boldsymbol{s}^{(i-1,j'')}$. Coordinate systems are depicted in dark gray with $\langle W \rangle$ and $\langle R \rangle$ denoting the world and the robot coordinate system, respectively.

contains the sensory information required for the further processing steps of the trajectory controller: the robot's current snapshot $\boldsymbol{S}^{(i,j)}$, an odometry-based estimate of the robot's position

$$\hat{\boldsymbol{s}}_{\text{odo}}^{(i,j)} = \left(\hat{x}_{\text{odo}}^{(i,j)}, \hat{y}_{\text{odo}}^{(i,j)}\right)^\top, \tag{4.2}$$

and a visually determined estimate $\hat{\theta}_{\text{vis}}^{(i,j)}$ of the robot's orientation. Lanes and snapshots along a lane are counted by $i = 0, 1, \ldots, i_{\max}$ and $j = 0, 1, \ldots, j_{\max}$, respectively. The following paragraph will describe the information stored in the place nodes in more detail.

The panoramic snapshot $\boldsymbol{S}^{(i,j)}$ is acquired at the unknown snapshot position

$$\boldsymbol{s}^{(i,j)} = (x, y, \theta)^\top \tag{4.3}$$

and is obtained by capturing an omnidirectional camera image, unfolding it to a cylindrical image, and preprocessing it (section 4.4.3). As we build a topo-metric map relying on a holistic representation of places (section 3.3.1.1), we use the entire image to characterize the position $\boldsymbol{s}^{(i,j)}$. The odometry estimates $\hat{\boldsymbol{s}}_{\text{odo}}^{(i,j)}$ (figure 4.2, gray diamonds) are given w.r.t. the world coordinate system $\langle W \rangle$, which is defined as a right-handed coordinate system at the initial robot position with its x-axis being aligned with the robot's initial movement direction. Due to the inaccuracy of the robot's odometry, the estimates $\hat{\boldsymbol{s}}_{\text{odo}}^{(i,j)}$ can deviate from the true but unknown robot positions $\boldsymbol{s}^{(i,j)}$. Because of this, true robot positions $\boldsymbol{s}^{(i,j)}$ and corresponding odometry estimates $\hat{\boldsymbol{s}}_{\text{odo}}^{(i,j)}$ are strictly separated in the description of the proposed controller (section 4.3). The odometry estimates $\hat{\boldsymbol{s}}_{\text{odo}}^{(i,j)}$ are used for selecting neighboring snapshots (section 4.3.3.1) and for estimating the base of the triangle for triangulation (section 4.3.3.3). The visual orientation estimate $\hat{\theta}_{\text{vis}}^{(i,j)}$ is obtained by fusing the compass information obtained from local visual homing with the orientation estimates of the previous lane (section 4.3.4.2).

At the current stage, we do not add links to the topo-metric map because the proposed controller operates solely on the graph's nodes. Nevertheless, links could, for example, be added depending on the neighbors or triangulation sets computed by the trajectory controller (sections 4.3.3.1 and 4.3.3.2).

In contrast to related work on trajectory-based SLAM, where the robot's full pose is determined for visually correcting the estimate of the robot's state (section 3.6.3.2), we do not correct the estimates stored in the map at a later point in time, and we solely compute the robot's orientation $\hat{\theta}_{\text{vis}}$. We refer to the latter aspect as partial pose estimation. Although we do not compute the robot's full pose, we consider the map built by our navigation strategy to be a topo-metric map because all orientation estimates $\hat{\theta}_{\text{vis}}^{(i,j)}$ are computed w.r.t. an external reference coordinate system. In future work, the place nodes can be extended by further task-relevant information such as local obstacle information.

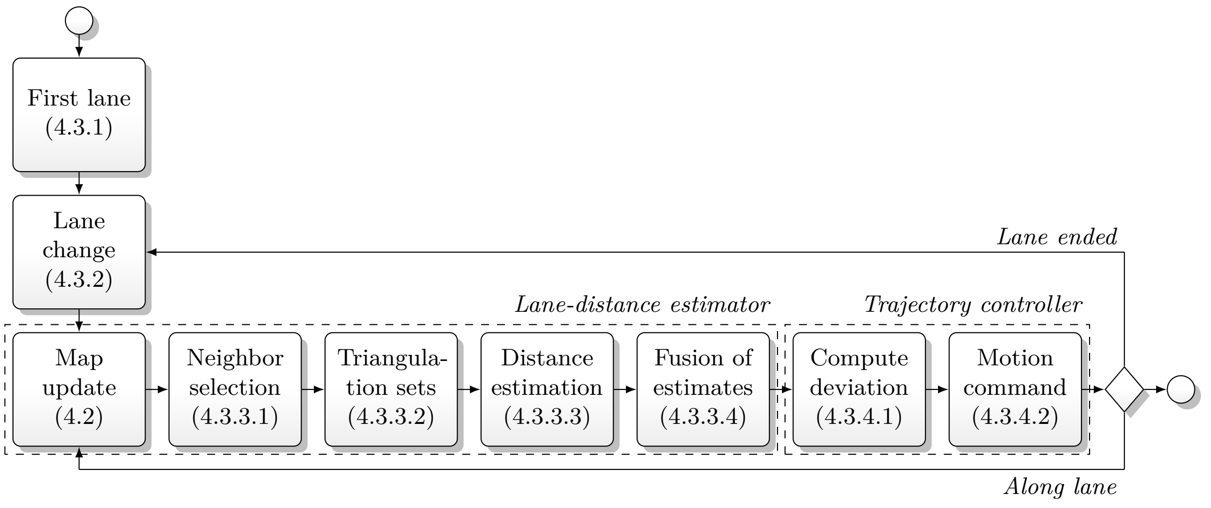

First lane (4.3.1)
Lane change (4.3.2)
Lane ended
Lane-distance estimator
Trajectory controller
Map update (4.2)
Neighbor selection (4.3.3.1)
Triangulation sets (4.3.3.2)
Distance estimation (4.3.3.3)
Fusion of estimates (4.3.3.4)
Compute deviation (4.3.4.1)
Motion command (4.3.4.2)
Along lane

Figure 4.3.: Flow chart of the proposed trajectory controller. The numbers in parentheses refer to the section describing each processing step in detail.

4.3. Trajectory Controller

In this section, the proposed trajectory controller is described in detail. Figure 4.3 visualizes the sequence of processing steps required to guide the robot along parallel and meandering lanes.

4.3.1. First Lane

While the robot moves along the first lane (lane counter $i = 0$), it successively adds place nodes $\mathcal{P}^{(0,j)}$ to its topo-metric map. The first lane is considered a special case because it does not have a predecessor, and therefore all processing steps requiring snapshots stored along the previous lane cannot be applied. These steps include (i) the lane-distance estimation and (ii) the estimation of the robot's current orientation. Both steps are required to compute motion commands for correcting the deviation from the desired inter-lane distance.

Instead of estimating and controlling the robot's distance to the previous lane, the first lane has to be kept straight by beacon aiming (e.g. by exploiting a beacon at the robot's docking station), by taking the bearing by means of local visual homing to former snapshots collected along the lane, or —as in the case of our experiments— by wall following. We are aware that wall following is not an option for keeping the first lane straight if the robot's environment is too cluttered, but we consider it to be a reasonable simplification to address the coverage of open spaces, which we think is (i) the more challenging aspect of cleaning-robot navigation and (ii) the more relevant aspect for proofing the method's feasibility. Since the walls used in our experiments are straight (section 4.4.1), all orientation estimates $\hat{\theta}_{\text{vis}}^{(0,j)}$ are assumed to be zero:

$$\hat{\theta}_{\text{vis}}^{(0,j)} = 0. \tag{4.4}$$

As a further simplification for our experiments, the first lane is ended after traveling for a fixed distance. If the proposed controller were integrated in a framework of further cleaning strategies, the first lane would end if an obstacle were detected or if the robot approached an already cleaned area. All subsequent lanes end if an obstacle is encountered or if the robot reaches the end of the previous lane. Since the proposed strategy relies on images stored along the previous lane for taking the bearing, the current lane cannot exceed its direct predecessor. In case the current lane is not blocked by an obstacle, reaching the end of the current lane is detected by simply counting the snapshots stored along this lane and stopping the robot if the number equals that of the previous

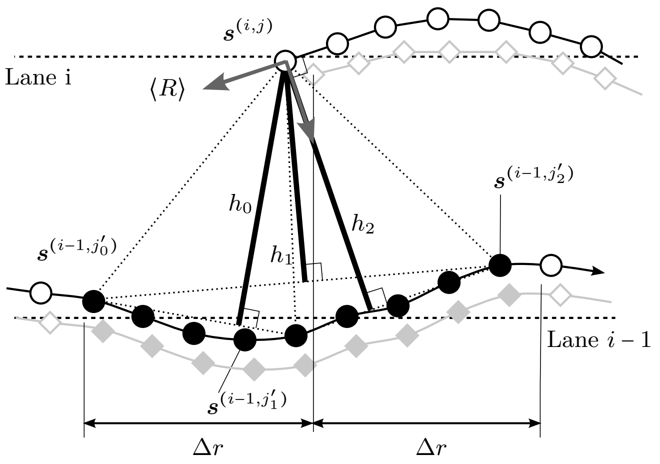


Figure 4.4.: Determining neighbors and triangulation sets. At the current robot position $\boldsymbol{s}^{(i,j)}$, a set $\mathcal{N}^{(i,j)}$ of neighbors (filled circles) along the previous lane is computed. The search radius Δr influences how many snapshots are selected as neighbors according to equation (4.5). Among all possible triangles with apex $\boldsymbol{s}^{(i,j)}$, three different triangles (dotted lines), referred to as triangulation sets $\mathcal{T}_k$, are used. The triangles' heights h_k are then used to compute an estimate $\hat{h}$ of the robot's current distance to the previous lane. The robot coordinate system $\langle R \rangle$ is depicted in dark gray.

lane. In principle, the snapshot counter j could be used to estimate the robot's traveled distance along the current lane or for deriving the robot's full position. However, because this estimate relied not on external sensor measurements, we expect it to be rather imprecise and we would rely on the robot's odometry instead.

4.3.2. Lane Changes

At the end of each lane, the robot performs a lane change. For the lane change, the robot turns by 90°, moves forward for the inter-lane distance Δl, turns again by 90°, and starts a new lane. The rotation of the robot is controlled by the visual compass method proposed by Zeil, Hoffmann, and Chahl [718]. While rotating, the robot continuously acquires images which are compared to a reference image acquired immediately before rotating. During the lane change, no place nodes are added to the topo-metric map, and the robot solely relies on its odometry without using any visual navigation method. In order to achieve optimal coverage, the inter-lane distance Δl should be chosen as the width of the robot's cleaning unit.

4.3.3. Lane-Distance Estimator

Along the second and all subsequent lanes (lane counter $i \geq 1$), the robot not only adds place nodes $\mathcal{P}^{(i,j)}$ to the map, but also computes the distance estimate $\hat{h}$ to the previous lane whenever a new node is added to the map. The processing steps required for this purpose will be described in the following.

4.3.3.1. Selection of Neighbors

After adding the current place node $\mathcal{P}^{(i,j)}$ to the map, its direct neighbors taken along the previous lane $i-1$ (figure 4.4, filled circles) are determined. These include the set

$$\mathcal{N}^{(i,j)} = \left\{ \mathcal{P}^{(i-1,j')} \mid \operatorname{abs}\left(\hat{x}_{\text{odo}}^{(i,j)} - \hat{x}_{\text{odo}}^{(i-1,j')} \right) < \Delta r \right\}, \tag{4.5}$$

i.e. the place nodes $\mathcal{P}^{(i-1,j')}$ taken along the previous lane for which the distance between $\hat{x}_{\text{odo}}^{(i,j)}$ and $\hat{x}_{\text{odo}}^{(i-1,j')}$ is smaller than Δr. For sake of simplicity, we only consider snapshots stored along the previous lane for the selection of neighbors. Since we assume that the robot initially moves into the positive x-direction of the world coordinate system $\langle W \rangle$ (section 4.2), the selection relies solely on the x-components of the position estimates $\hat{\boldsymbol{s}}$ obtained by the robot's odometry. Due to being only a coarse search for neighboring snapshots, the selection is the only part of the proposed algorithm

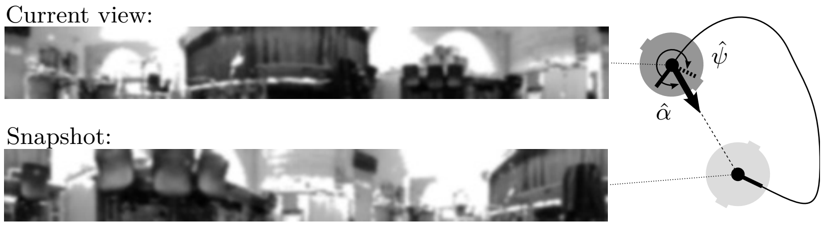


Figure 4.5.: Local visual homing. The robot moves from its start position (light gray) to its current position (gray). By comparing the current view (CV) with the snapshot, i.e. the camera image acquired at the start position (SS), estimates of the home direction $\hat{\alpha}$ as well as of the relative orientation change $\hat{\psi}$ between the considered snapshots (also referred to as visual compass) can be computed w.r.t. the robot's current orientation (figure: black thick bars). The panoramic images show the laboratory in which experiments were conducted. Images were obtained with the omnidirectional vision setup and the parameters used for the experiments (section 4.4). Since parameter optimizations for the used 2D warping methods have shown that the method performs best with a certain level of low-pass filtering (Möller [447] and Möller, Krzykawski, and Gerstmayr [451]), the images appear rather blurred.

relying on absolute position estimates obtained by the robot's odometry (light-gray diamonds). Alternatively, a snapshot counter along the lanes could be used. The choice of the search radius Δr influences how many snapshots are selected as neighbors and by this means also the accuracy of the estimate $\hat{h}_k$ of the inter-lane distance computed in the subsequent processing steps (sections 4.3.3.2 to 4.3.3.3). If Δr is chosen too small, the snapshots considered along the previous lane for taking the bearing are too close together, and the quality of the resulting home vectors can be decreased due to local properties of the environment. Otherwise, if Δr is chosen too large, the estimate $\hat{h}_k$ becomes more sensitive to small deviations from the true home vector due to the tangents being involved in equation equation (4.23). According to our experience, good results are obtained if the resulting triangles are approximately right-angled. This observation is in line with the parameter optimization in the preliminary study by Dr. Sven Kreft [342] suggesting to use $\Delta r = 45\,\mathrm{cm}$.

4.3.3.2. Triangulation Sets

Based on the set $\mathcal{N}^{(i,j)}$ of neighbors, a subset of place nodes is selected, which will in the following step be used to compute the robot's current distance to the previous lane. Among all neighboring place nodes in $\mathcal{N}^{(i,j)}$, we use the outermost nodes $\mathcal{P}^{(i-1,j'_0)}$ and $\mathcal{P}^{(i-1,j'_2)}$ as well as an intermediate place node $\mathcal{P}^{(i-1,j'_1)}$ with

$$j'_0 = \arg\min_{j'} \left\{ \hat{x}_{\text{odo}}^{(i-1,j')} \in \mathcal{N}^{(i,j)} \right\}, \tag{4.6}$$

$$j'_2 = \arg\max_{j'} \left\{ \hat{x}_{\text{odo}}^{(i-1,j')} \in \mathcal{N}^{(i,j)} \right\}, \text{and} \tag{4.7}$$

$$j'_1 = \text{floor}\left(\frac{j'_0 + j'_2}{2} \right) \tag{4.8}$$

with the floor function computing the largest index not greater than the function's argument. The snapshots $\boldsymbol{S}^{(i-1,j'_0)}$, $\boldsymbol{S}^{(i-1,j'_1)}$, and $\boldsymbol{S}^{(i-1,j'_2)}$ associated with these place nodes are in subsequent steps used to take the bearing from the current robot position $\boldsymbol{s}^{(i,j)}$ to each of the former robot positions $\boldsymbol{s}^{(i-1,j'_k)}$, at which these snapshots were acquired. For this purpose, three homing angles $\hat{\alpha}_k$ and changes of the orientation $\hat{\psi}_k$ are computed by local visual homing (figure 4.5 and section 3.5)

$$(\hat{\alpha}_k, \hat{\psi}_k)^\top = \text{lvh}\left(\boldsymbol{S}^{(i,j)}, \boldsymbol{S}^{(i-1,j'_k)} \right) \text{ with } k \in \{0,1,2\} \tag{4.9}$$

and lvh describing local visual homing as a function of the two snapshots $\boldsymbol{S}^{(i,j)}$ and $\boldsymbol{S}^{(i-1,j'_k)}$. Since the results of local visual homing have to be considered as a noisy and potentially error-prone

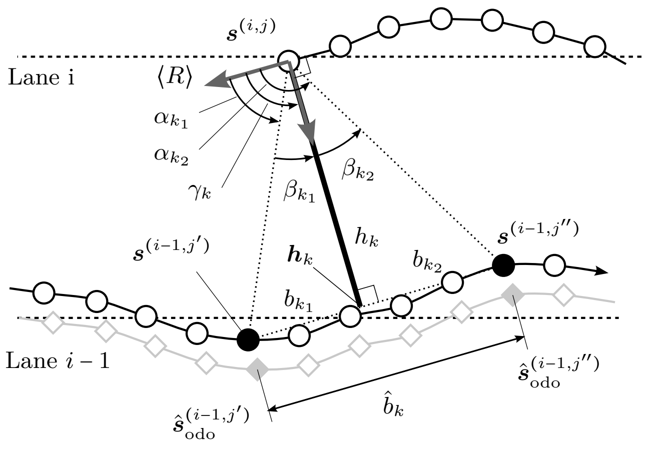


Figure 4.6.: Visual triangulation used to estimate the robot's current distance to the previous lane. An estimate $\hat{h}_k$ of the triangle's height can be obtained because the two angles α_{k_1} and α_{k_2} can be determined by visual homing, and because the base length $b_k = b_{k_1} + b_{k_2}$ can be approximated by the relative distance $\hat{b}$ based on the robot's odometry. For details please refer to section 4.3.3; dark-gray arrows depict the robot coordinate system $\langle R\rangle$.

measurement rather than an exact computation (e.g. [451]), we refer to them as estimates $\hat{\alpha}_k$ and $\hat{\psi}_k$.

Together with the current place node $\mathcal{P}^{(i,j)}$, the place nodes $\mathcal{P}^{(i-1,j_0')}$, $\mathcal{P}^{(i-1,j_1')}$, and $\mathcal{P}^{(i-1,j_2')}$ are combined to three triangulation sets (figure 4.4, dotted lines)

$$\mathcal{T}_0 = \left\{\mathcal{P}^{(i,j)}, \mathcal{P}^{(i-1,j_0')}, \mathcal{P}^{(i-1,j_1')}\right\}, \tag{4.10}$$

$$\mathcal{T}_1 = \left\{\mathcal{P}^{(i,j)}, \mathcal{P}^{(i-1,j_0')}, \mathcal{P}^{(i-1,j_2')}\right\}, \text{and} \tag{4.11}$$

$$\mathcal{T}_2 = \left\{\mathcal{P}^{(i,j)}, \mathcal{P}^{(i-1,j_1')}, \mathcal{P}^{(i-1,j_2')}\right\}. \tag{4.12}$$

Each triangulation set will in the following step be used to compute an estimate $\hat{h}_k$ (with $k \in \{0,1,2\}$) of the robot's distance h to the previous lane (thick black lines). In case the robot's true trajectory deviates from the ideal trajectory with its parallel and straight lanes (dashed lines), different triangulation sets will also result in different estimates $\hat{h}_k$. In order to improve the estimate $\hat{h}$ of the robot's true distance h to the previous lane, the estimates $\hat{h}_k$ of several triangles will be computed and fused. Using three triangulation sets has proven to be a good trade-off between computational complexity and estimation accuracy because experiments have shown that further increasing the number of considered triangles could not improve the performance of the algorithm. Nevertheless, further triangulation sets could be determined by using a similar scheme for more than three triangles.

4.3.3.3. Distance Estimation

For each triangulation set $\mathcal{T}_k$ ($k \in \{0,1,2\}$) determined in the previous step, an estimate $\hat{h}_k$ of the robot's true distance h to the previous lane is computed. The following derivation relies on the fact that a triangle is completely specified by two angles and the length of one of its sides (figure 4.6). In our case, we use the angles

$$\beta_{k_1} = \measuredangle\left(\boldsymbol{s}^{(i-1,j')}, \boldsymbol{s}^{(i,j)}, \boldsymbol{h}_k\right) \text{ and} \tag{4.13}$$

$$\beta_{k_2} = \measuredangle\left(\boldsymbol{h}_k, \boldsymbol{s}^{(i,j)}, \boldsymbol{s}^{(i-1,j'')}\right), \tag{4.14}$$

where $\boldsymbol{h}_k$ is the foot of the triangle's height. The base is the line segment bounded by the snapshot positions $\boldsymbol{s}^{(i-1,j')}$ and $\boldsymbol{s}^{(i-1,j'')}$. As side length, we rely on the triangle's base length b_k. With these quantities, the triangle's height h_k can be obtained by resolving the relation

$$b_k = b_{k_1} + b_{k_2} = h_k(\tan\beta_{k_1} + \tan\beta_{k_2}) \tag{4.15}$$

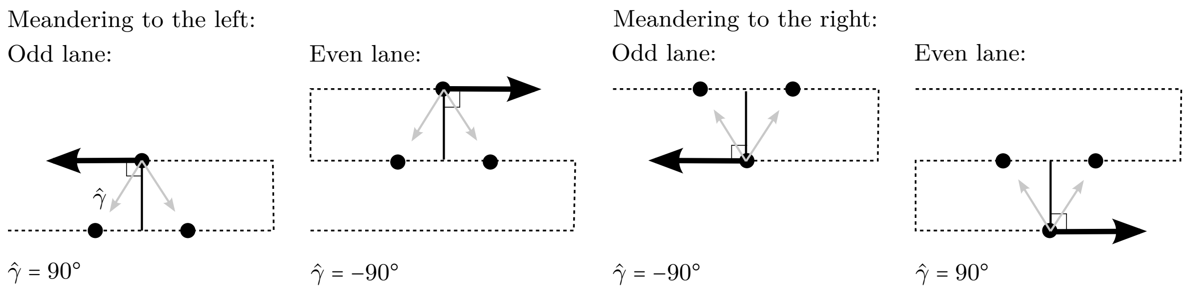


Figure 4.7.: $\hat{\gamma}$ depending on direction of travel and direction of meandering. The sign of the angle $\hat{\gamma}$ is chosen depending on the direction of meandering (i.e. the direction of the first turn at the end of the first lane) and on the robot's current direction of travel (thick black arrow). Dashed lines depict the robot's trajectories, black filled circles snapshot positions, and light-gray arrows the bearing direction computed by local visual homing.

for h_k:

$$h_k = \frac{b_k}{\tan \beta_{k_1} + \tan \beta_{k_2}}. \tag{4.16}$$

However, none of these quantities are known or can be derived from the available information, and thus further assumptions are required for approximating them.

As angular information, the homing angles α_{k_1} and α_{k_2} between the robot's current heading and the snapshot positions $\boldsymbol{s}^{(i-1,j')}$ and $\boldsymbol{s}^{(i-1,j'')}$ can be obtained by taking the bearing from the robot's current position $\boldsymbol{s}^{(i,j)}$ to the triangle's base points $\boldsymbol{s}^{(i-1,j')}$ and $\boldsymbol{s}^{(i-1,j'')}$ (equation (4.9)). The homing angles α_{k_1} and α_{k_2} are related to β_{k_1} and β_{k_2} by the unknown angle γ_k between the robot's heading and the triangle's height h_k:

$$\beta_{k_1} = \gamma_k - \alpha_{k_1} \text{ and} \tag{4.17}$$

$$\beta_{k_2} = \alpha_{k_2} - \gamma_k. \tag{4.18}$$

By assuming $\gamma_k = \hat{\gamma} \in \{-90°, 90°\}$ with the sign depending on the robot's current direction of travel and the direction of meandering (figure 4.7) and by considering that taking the bearing only provides estimates $\hat{\alpha}_{k_1}$ and $\hat{\alpha}_{k_2}$ of the exact homing angles α_{k_1} and α_{k_2}, we can approximate β_{k_1} and β_{k_2} by

$$\hat{\beta}_{k_1} = \hat{\gamma} - \hat{\alpha}_{k_1} \text{ and} \tag{4.19}$$

$$\hat{\beta}_{k_2} = \hat{\alpha}_{k_2} - \hat{\gamma}. \tag{4.20}$$

The assumption $\hat{\gamma} = \pm 90°$ holds exactly if and only if the robot is moving parallel to its previous lane as depicted in figure 4.6 and for the triangulation set $\mathcal{T}_2$ in figure 4.4. In other cases, the assumption is violated (triangulation sets $\mathcal{T}_0$ and $\mathcal{T}_1$ in figure 4.4), and deviations from the robot's desired lane are likely to occur.

Furthermore, the true base length b_k is unknown because the positions of image acquisitions $\boldsymbol{s}^{(i-1,j')}$ and $\boldsymbol{s}^{(i-1,j'')}$ are unknown. As the robot's odometry obtains good estimates for distances between snapshots taken along a lane, we approximate b_k by the Euclidean distance

$$\hat{b}_k = \left\| \hat{\boldsymbol{s}}_{\text{odo}}^{(i,j')} - \hat{\boldsymbol{s}}_{\text{odo}}^{(i,j'')} \right\| \tag{4.21}$$

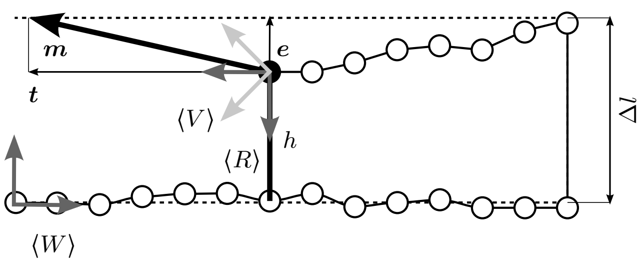


Figure 4.8.: Computation of the motion vector. The proposed controller computes a motion vector $\boldsymbol{m}$ which reduces the deviation $e = \|\boldsymbol{e}\|$ from the robot's current distance h to the previous lane and the desired lane distance Δl. The robot is supposed to return to the desired lane within some moving target distance $t = \|\boldsymbol{t}\|$. To determine the wheel speeds for controlling the robot, $\boldsymbol{m}$ is transformed from world coordinates $\langle W \rangle$ (dark-gray arrows) to the robot coordinate system $\langle R \rangle$ (dark-gray arrows) and further to an auxiliary coordinate system $\langle V \rangle$ (light-gray arrows).

between the odometry estimates $\hat{\boldsymbol{s}}_{\text{odo}}^{(i-1,j')}$ and $\hat{\boldsymbol{s}}_{\text{odo}}^{(i-1,j'')}$. Based on these assumptions, the estimate $\hat{h}_k$ for the robot's current distance h to the previous lane is then obtained by

$$\hat{h}_k = \frac{\hat{b}_k}{\tan\hat{\beta}_{k_1} + \tan\hat{\beta}_{k_2}} \tag{4.22}$$

$$= \frac{\hat{b}_k}{\tan(\hat{\gamma} - \hat{\alpha}_{k_1}) + \tan(\hat{\alpha}_{k_2} - \hat{\gamma})}. \tag{4.23}$$

4.3.3.4. Fusion of Several Estimates

To fuse the estimates $\hat{h}_k$ of the robot's current distance to the previous lane, the median $\hat{h}$ over all estimates is computed:

$$\hat{h} = \operatorname*{median}_k \hat{h}_k \text{ with } k \in \{0, 1, 2\}. \tag{4.24}$$

The result is used by the control algorithm (section 4.3.4) to derive a motion command keeping the robot on a course parallel to the previous lane.

4.3.4. Trajectory Controller

The trajectory controller uses the estimate $\hat{h}$ of the robot's current distance h to its previous lane to keep the robot at the desired distance Δl from the previous lane. To reduce the deviation from the desired lane, a motion vector $\boldsymbol{m}$ is computed, which is mapped to wheel speeds in order to control the robot. Keeping the robot's distance to the previous lane constant implies that the robot's desired lane is specified w.r.t. its previous lane.

4.3.4.1. Motion Vector

The motion vector $\boldsymbol{m}$ is supposed to compensate the deviation

$$e = \Delta l - \hat{h}, \tag{4.25}$$

and is defined as the vector sum

$$^{\langle W \rangle}\boldsymbol{m} = \boldsymbol{e} + \boldsymbol{t} = \begin{pmatrix} 0 \\ s_e \cdot e \end{pmatrix} + \begin{pmatrix} s_t \cdot t \\ 0 \end{pmatrix} \tag{4.26}$$

of the deviation vector $\boldsymbol{e}$ and the moving target vector $\boldsymbol{t}$ (figure 4.8). The vector $\boldsymbol{e}$ is chosen parallel to the y-axis of the world coordinate system $\langle W \rangle$ pointing always into the direction compensating the deviation e. The moving target vector $\boldsymbol{t}$ points parallel to the x-axis of $\langle W \rangle$ into the current movement direction. Its length $t = \|\boldsymbol{t}\|$ is the moving target distance within which the robot is supposed to completely reduce the deviation from its desired lane. With small choices of t, the

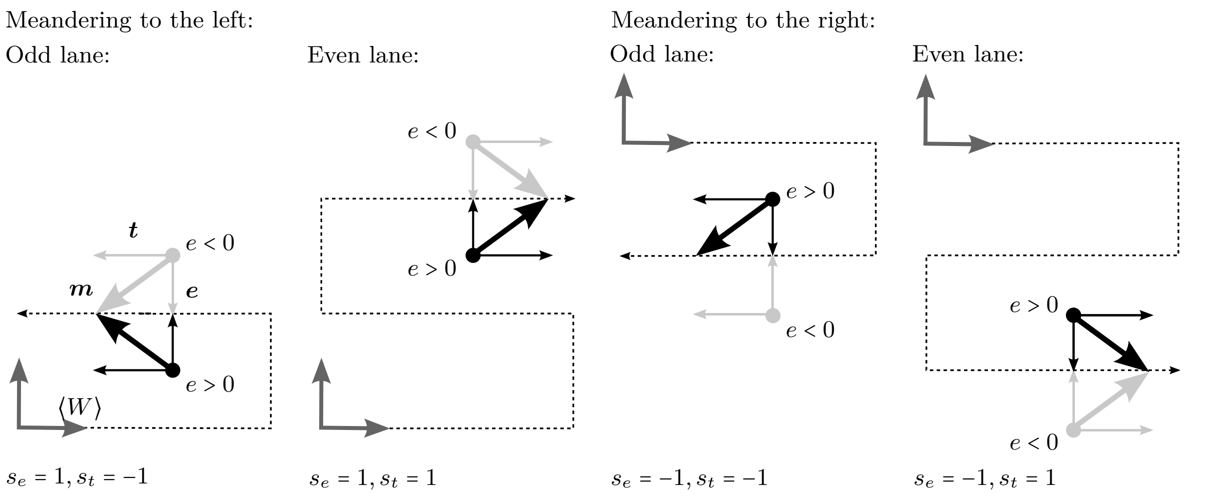


Figure 4.9.: Sign of the multipliers s_e and s_t depending on the directions of meandering and of the current lane. Compensating the deviation e from the desired inter-lane distance by computing a motion vector $\boldsymbol{m} = \boldsymbol{e} + \boldsymbol{t}$ (black and light-gray arrows) which guides the robot back to its desired trajectory (dashed line). The directions of $\boldsymbol{e}$ and $\boldsymbol{t}$ are adjusted by selecting the multipliers s_e and s_t (equation (4.26)) depending on the direction of the lane change at the end of the first lane (left or right) and on the direction of travel along the current lane (even or odd lane). The multipliers do not depend on the sign of the deviation e. The world coordinate system $\langle W \rangle$ is depicted by dark-gray arrows.

controller is more reactive and capable of quickly reducing the deviation; larger choices of t make the controller slower but result in smoother trajectories. The multipliers s_e and s_t, both from the set $\{-1, 1\}$, adjust the directions of the vectors $\boldsymbol{e}$ and $\boldsymbol{t}$ depending on the robot's direction of travel and its direction of meandering, i.e. the directions of its initial turn after finishing the first lane (figure 4.9).

4.3.4.2. Transformation to Wheel Speeds

In equation (4.26), the motion vector $\boldsymbol{m} = {}^{\langle W \rangle}\boldsymbol{m}$ was defined w.r.t. world coordinates $\langle W \rangle$. In order to map the motion vector $\boldsymbol{m}$ to wheel speeds correcting the robot's deviation from the desired lane distance Δl, ${}^{\langle W \rangle}\boldsymbol{m}$ has to be transformed from the world coordinate system $\langle W \rangle$ to the robot coordinate system $\langle R \rangle$ and further to an auxiliary system $\langle V \rangle$.

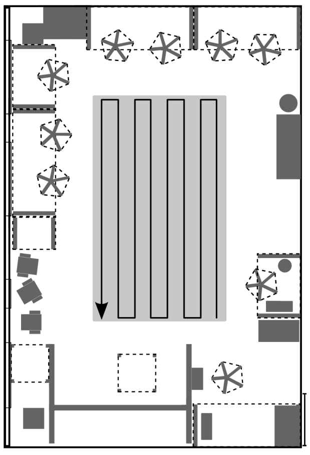
As the robot's position in world coordinates is unknown due to partial pose estimation, only the rotational component of the coordinate transformation from $\langle W \rangle$ to $\langle R \rangle$ can be computed. However, this is sufficient for the following processing steps, and we do not have to rely on further assumptions. To determine the rotation between world coordinate system $\langle W \rangle$ and robot coordinate system $\langle R \rangle$, an estimate of the robot's orientation $\hat{\theta}^{(i,j)}_{\text{vis}}$ is required. It is derived by angular averaging (textbook: [32])

$$\hat{\theta}^{(i,j)}_{\text{vis}} = \operatorname*{mean}_{k} \hat{\theta}^{(i,j)}_{\text{vis},k} \tag{4.27}$$

the orientation estimates $\hat{\theta}^{(i,j)}_{\text{vis},k}$ ($k \in \{0, 1, 2\}$) obtained for each of the computed home vectors. These are obtained by the angular summation

$$\hat{\theta}^{(i,j)}_{\text{vis},k} = \psi_k + \hat{\theta}^{(i-1,j'_k)}_{\text{vis}} \tag{4.28}$$

of the relative change of orientation ψ_k computed in equation (4.9) and the vision-based orientation estimate $\hat{\theta}^{(i-1,j'_k)}_{\text{vis}}$ associated with place node $\mathcal{P}^{(i-1,j'_k)}$ taken along the previous lane. Based on the


(1) Experiment 1

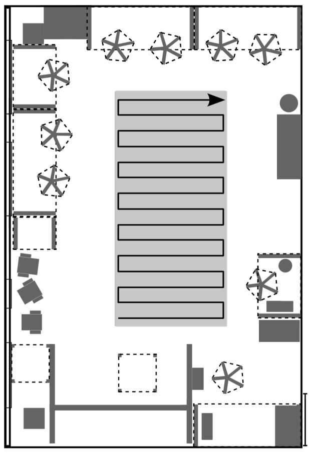

(2) Experiment 2

Figure 4.10.: Areas ideally covered by the robot experiment 1 and 2 (subfigures (1) and (2), respectively). Depicted are ground-level objects (e.g. legs of chairs and desks or closets; dark gray areas), obstacles above ground level (e.g. surfaces of chairs or desks; dashed lines), the ideal cleaning trajectories (black arrows), and the area ideally covered by the cleaning runs (light gray areas). The area covered by the cleaning trajectories is approximately the field of view of the visual tracking system (section 4.4.5); the light-gray area was used as reference for computing the cleaning performance (section 4.4.6). The measuring line at the lower right corner shows a length of 1 m.

orientation estimate $\hat{\theta}^{(i,j)}_{\text{vis}}$, the vector ${}^{\langle R\rangle}\boldsymbol{m}$ is obtained by rotating ${}^{\langle W\rangle}\boldsymbol{m}$ by $-\hat{\theta}^{(i,j)}_{\text{vis}}$ around the robot's z-axis.

To transform the motion vector ${}^{\langle R\rangle}\boldsymbol{m}$ into velocities $\boldsymbol{v} = (v_L, v_R)$ for the left and right wheel of the robot, the mapping proposed by Möller [446] is applied. Therefore, the movement vector ${}^{\langle R\rangle}\boldsymbol{m}$ is further transformed into the coordinate system $\langle V\rangle$ which is rotated by -45° around the z-axis of the robot coordinate system $\langle R\rangle$. To obtain wheel speeds, the resulting vector ${}^{\langle V\rangle}\boldsymbol{m}$ is normalized and scaled with the desired velocity v:

$$\boldsymbol{v} = \begin{pmatrix} v_L \\ v_R \end{pmatrix} = \frac{v}{\|{}^{\langle V\rangle}\boldsymbol{m}\|} \cdot {}^{\langle V\rangle}\,\boldsymbol{m}. \tag{4.29}$$

The resulting velocities v_L and v_R are passed to the robot's motion controller and kept constant until the next place node is added.

4.4. Experiments and Setup

In order to test the proposed navigation strategy, we conducted experiments with a custom-built cleaning robot. Section 4.4.1 describes the experimental procedure and the parameters used for the experiments. For computing compass and bearing information, the min-warping method (section 4.4.2) is applied. Section 4.4.3 briefly introduces the custom-built cleaning robot. To demonstrate the capabilities of the proposed controller, the motion commands were disturbed with a strong systematic error to be compensated by the controller (section 4.4.4). The experiments were recorded with a vision-based tracking system (section 4.4.5), and the resulting trajectories were analyzed qualitatively and quantitatively (section 4.4.6).

4.4.1. Procedure and Used Parameters

This section introduces the experimental procedure and the parameters used for our experiments. In order to keep this description compact, details or discussions on different aspects only briefly mentioned in this section can be found in sections 4.4.2 to 4.4.6. We performed cleaning runs from two different start positions in our lab (figure 4.10). In experiment 1, the target trajectory consisted of eight lanes each of 4 m length; in experiment 2, it consisted of 15 lanes with a lane length of 2 m. In both experiments, the robot was meandering to the left, and a total of 10 trials per experiment were recorded. The size and the position of the workspace within our lab were limited by the field of view of our visual tracking system, which was used to record the robot's trajectories (section 4.4.5).

Due to properties of the tracking system, experiments had to be conducted under constant and diffuse illumination conditions and in a static environment (see section 4.4.5 for details).

The robot's motion controller was disturbed by a systematic error of 5 %, which had to be compensated by the controller in order to keep the robot parallel to the previous lane (section 4.4.4). We consider the specific choice to be a reasonable error because its effect is clearly visible if the robot is moving relying solely on its odometry (figure 4.14 and video [S2]) and because —if moving under visual control— the performance is not considerably decreased in comparison to a smaller or a zero error (section 4.5.1 and the videos [S3] and [S4] with and without odometry disturbance, respectively). Errors larger than 10 % can cause the method to fail. Half of the trials were disturbed with a systematic error causing a trajectory curved to the left; the other half was performed with an error causing a trajectory curved to the right. For data analysis (section 4.4.6), the results for both types of error were pooled because the results are independent of the direction of the systematic error.

Along a cleaning lane, the robot was continuously moving with $v = 7.5\,\text{cm/s}$, the inter-snapshot distance was $\Delta s = 10\,\text{cm}$, the moving-target distance was $t = 45\,\text{cm}$, and the radius of the neighbor search was $\Delta r = 45\,\text{cm}$. As our robot (section 4.4.3) is not equipped with a suction unit, we used an inter-lane distance of $\Delta l = 30\,\text{cm}$, which is approximately the robot's diameter. Since we observed in simulation studies that the system achieves similar performance for different parameter combinations, we only present results for this parameter set (rather then presenting data for systematically varying the moving target distance t, the search radius r, the odometry error k, and the number of triangulation pairs).

The first lane was kept straight by following an artificial wall relying on the robot's IR range sensors. The artificial wall has a height of approximately 9 cm and is not visible for the robot because it is completely below the horizon of its omnidirectional vision system (see figure 4.5 for omnidirectional images acquired during our experiments). An artificial wall was used (i) because our lab is too cluttered and does not offer a free wall sufficiently long for our experiments (figure 4.10) and (ii) because the experiments had to be conducted in an area which is observable by the used tracking system (section 4.4.5). This restriction (see also the discussion in section 4.3.1) will be lifted in future work on full-fledged cleaning strategies (section 7.3.2).

For all experiments, min-warping with compass acceleration was used as homing method (section 4.4.2). In the search process, the home direction α and the orientation change ψ were varied in discrete steps of 5°. The compass acceleration restricted the search space to 30 % of the possible orientation changes, image columns were compared by the Euclidean distance, and we used 9 different scale planes in the range $[0.2, 1.8]$.

4.4.2. Min-Warping With Compass Acceleration

Among all homing methods (section 3.5.2), image-warping methods [190, 447, 451] have proven to be accurate and robust. Furthermore, warping methods not only compute an estimate $\hat{\alpha}$ of the home direction but also of the azimuthal orientation difference $\hat{\psi}$ between the two images (visual compass; figure 4.5). For the proposed navigation strategy, we use a variant of 2D-warping called min-warping with compass acceleration because it offers a good trade-off between homing accuracy and computational complexity (Möller, Krzykawski, and Gerstmayr [451]). The method uses an explicit search for discrete combinations of the home direction α and the orientation change ψ. Depending on α and ψ, the shift and scale change of an image feature (in the case of min-warping the complete image column of a panoramic image) can be computed. The search space of orientation changes can be restricted to a small fraction of likely orientations by applying a visual compass method implicitly contained in the min-warping method (in Möller, Krzykawski, and Gerstmayr [451] referred to as compass acceleration). The parameters α and ψ resulting in the best match between snapshot and current view columns are used as estimates $\hat{\alpha}$ and $\hat{\psi}$ of the

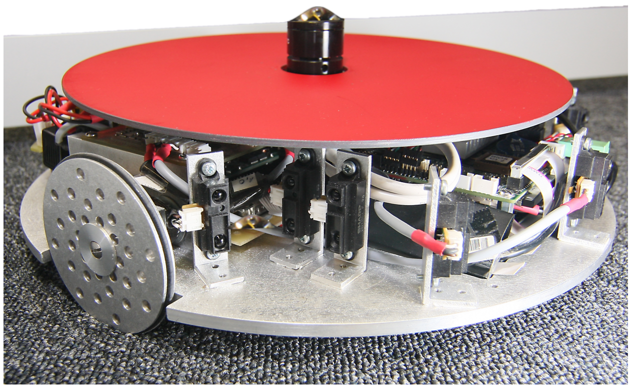


Figure 4.11.: Custom-built cleaning robot with panoramic annular lens (PAL) as omnidirectional-vision sensor. For recording the robot's trajectory, the robot is equipped with a colored disk (here, the robot is depicted with a red disk) which is used as tracking target by the visual tracking system (section 4.4.5 and figure 4.15). In the shown photo, the tracking target shields the motors, the motor controllers, the on-board computer, the batteries, and the camera [D9]. The chassis is supported by a ball caster not visible in the photo. Visible are the panoramic annular lens (PAL, top center; [D16]) and IR distance sensors (black vertical blocks, Sharp GP2D12 and GP2D120, [D12, D13]). The artificial wall used for the experiments is visible in the image's background. The robot was mostly designed and built by Klaus Kulitza; Martin Krzykawski was responsible for developing the software required for robot communication and control. Photo by Lorenz Hillen. Figure best viewed in color.

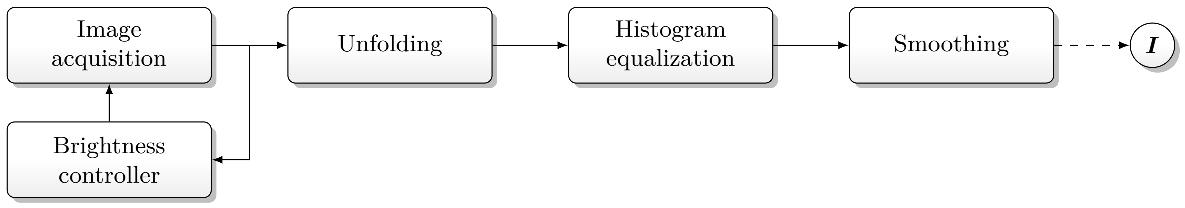


Figure 4.12.: Image preprocessing steps. All preprocessing operations are executed on the robot's on-board computer. The resulting panoramic image $\boldsymbol{I}$ is transferred to the external host computer via wireless network connection (dashed arrow) where the further processing steps are executed.

home direction and of the orientation change. Since warping methods use the entire image to derive the home vector and compass estimates, they are well suited for navigation methods operating on graph-based maps (with or without position information) and a holistic representation of places (sections 3.6.3.2 and 3.6.4.1).

4.4.3. Custom-Built Cleaning Robot

Experiments were conducted with a custom-built differential-drive robot (figure 4.11). With a diameter of 33 cm and a height of 12 cm, its dimensions are comparable to those of commercially available cleaning robots for domestic usage (section 2.2.1). In contrast to such robots, our robot is not equipped with a cleaning unit. For obstacle detection and wall following, the robot is equipped with IR distance sensors (Sharp GP2D12 and GP2D120, [D12, D13]). As an omnidirectional vision sensor, the robot relies on a camera (IDS Imaging UI-2220SE-M, [D8]) with a panoramic annular lens (PAL, Tateyama S25G2817-27C [D16], patent: [251]). The used omnidirectional vision setup is also depicted in figure 3.7.2. Since the PAL is much more compact than standard catadioptric sensors, it sticks out of the housing by only 3 cm.

Figure 4.12 visualizes the image preprocessing steps. The panoramic camera images were unfolded to panoramic images by applying a custom-developed model-free mapping method (Krzykawski [345]). Unfolding included a histogram equalization followed by low-pass filtering (binomial filter with kernel size 5). The resulting images were sized 360×48 pixels and covered a vertical field of view from $0°$ to $38°$ above the horizon (see figure 4.5 for two example images). Image unfolding was done on the robot's on-board PC (IEI Technology PM-US15W-Z530-R10 with an Intel Atom Z530 CPU, [D10]), and the unfolded images were transmitted to an external host computer (laptop [D4] with an Intel Core i7 920XM CPU and 4 GB of RAM) via wireless network connection. There, the computations required for the proposed navigation strategy (namely computing three home

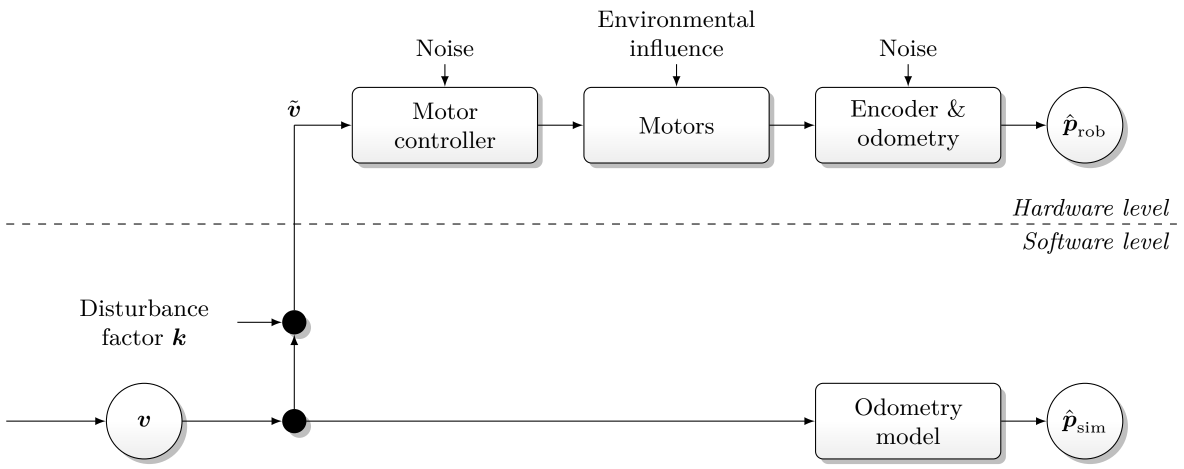


Figure 4.13.: Artificial disturbance of the robot's motion controller. The velocities $\boldsymbol{v}$ passed to the robot's motion controller are disturbed by a systematic error k resulting in biased velocities $\tilde{\boldsymbol{v}}$ causing the robot to move on a circular path. As the robot's on-board odometry can measure this disturbance, the position estimate $\hat{\boldsymbol{p}}_{\text{rob}}$ should not be used for the trajectory controller. Instead, the robot's on-board odometry is bypassed by a secondary odometry computing the robot's position $\hat{\boldsymbol{p}}_{\text{sim}}$ based on unbiased velocities $\boldsymbol{v}$.

vectors, estimating the inter-lane distance, and deriving a motion command) were executed, and the resulting motion command was sent back to the robot. The image capturing and preprocessing as well as the client-server architecture was written by Martin Krzykawski. Among these processing steps, computing a single home vector requires approximately 70 ms (Möller, Krzykawski, and Gerstmayr [451]) allowing for real-time control of the robot. In comparison to local visual homing, the other operations of the visual controller are negligible. The off-board computations and the client-server architecture were necessary because the proposed method was implemented relying on the prototyping software framework maintained by our group. Nevertheless, because our method was designed keeping in mind the limited computational power of autonomous cleaning robots, we expect it to be executable in real-time on a robot's on-board computer if the current implementation is optimized for this computer hardware. Although we are aware of its importance for a fully autonomous cleaning robot, such an implementation is left for future work.

4.4.4. Systematic Error

In order to verify that the visual trajectory controller (section 4.3.4) can keep the robot on the desired trajectory, the robot's motion controller is in all experiments biased by a strong systematic error. The error simulates differences in the robot's wheel diameters causing the robot to move on a circular path, which has to be compensated by the proposed navigation strategy. In case of differences in the wheel diameters, the robot's odometry cannot measure that the robot is moving on a curved trajectory. Thus, the resulting position estimates $\hat{\boldsymbol{p}}_{\text{rob}}$ lie on a straight line.

Adding a systematic error to the robot's desired wheel speeds $\boldsymbol{v} = (v_L, v_R)^\top$ by disturbing the velocities with a constant disturbance factor k proportional to the systematic error

$$\tilde{\boldsymbol{v}} = \begin{pmatrix} \tilde{v}_\text{L} \\ \tilde{v}_\text{R} \end{pmatrix} = \begin{pmatrix} \left(1 + \frac{k}{2}\right) v_\text{L} \\ \left(1 - \frac{k}{2}\right) v_\text{R} \end{pmatrix} \tag{4.30}$$

causes the robot to move on a circular path. With this equation, we assure that the error symmetrically influences both wheels without influencing the overall robot velocity. However, the robot's on-board odometry can measure that the robot moves on a curved path. Thus, relying on the odometry estimates $\hat{\boldsymbol{p}}_{\text{rob}}$ would influence the controller, e.g. when the base length $\hat{b}$ of the considered triangle is estimated (equation (4.21)).

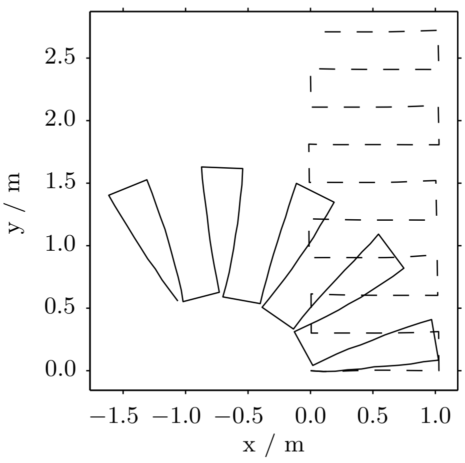


Figure 4.14.: Influence of disturbed odometry. Trajectory resulting from driving the robot along meandering lanes solely based on its disturbed odometry (i.e. without visual correction). The continuous black line depicts the robot's true course as recorded by the visual tracking system; the dashed line depicts the position estimates obtained by the disturbed odometry. A supplemental video visualizing the depicted effects is available for download [S2].

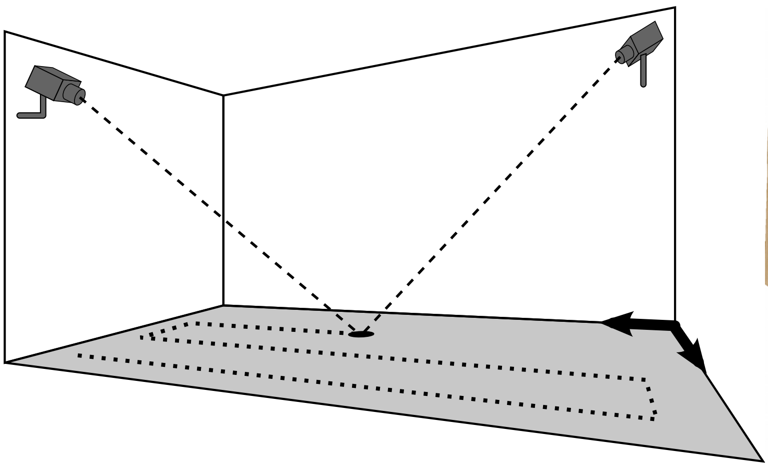
In order to eliminate these effects, we bypass the robot's on-board odometry (figure 4.13). For this purpose, a position estimate $\hat{\boldsymbol{p}}_{\mathrm{sim}}$ is computed by applying a standard odometry model of differential-drive robots (e.g. section 5.2.4 of Siegwart, Nourbakhsh, and Scaramuzza [586]). As the computation of this estimate is based on the unbiased wheel speeds $\boldsymbol{v}$, the simulated odometry cannot measure the influences of the disturbance k. Whenever the odometry is read out, the request is not directed to the robot's on-board odometry, but the simulated odometry is updated, and its position estimate $\hat{\boldsymbol{p}}_{\mathrm{sim}}$ is returned. Since the simulated odometry does not consider environmental influences (such as motor noise or wheel slippage), and since the simulated odometry assumes instantaneous changes of the wheel speeds, the position estimates of the simulated and the robot's on-board odometry without systematic error ($k = 0$) differ. However, we think that these differences are negligible.

Figure 4.14 and the video [S2] visualize the effects of a 5 % systematic error (i.e. $k = \pm 0.05$) as used for the experiments (section 4.4.1). In the shown case, the error was $k = -0.05$, thus simulating that the robot's right wheel has a larger diameter than the left one. The robot was driven along meandering lanes of length 1 m without visual correction. As the robot's true trajectory (continuous black line) is curved whereas the trajectory obtained from the robot's odometry (dashed line) is parallel and meandering, the approach described in this section simulates differences in the diameter of the robot's wheels. In case the experiments reveal that the robot is guided along parallel lanes, these results therefore have to be due to the visual trajectory controller compensating for the strong systematic error introduced by k. Please note that these errors were only introduced for testing the trajectory controller; future work focusing on more elaborated cleaning strategies (section 4.7) will rely on the robot's on-board odometry.

4.4.5. Passive Visual Tracking System

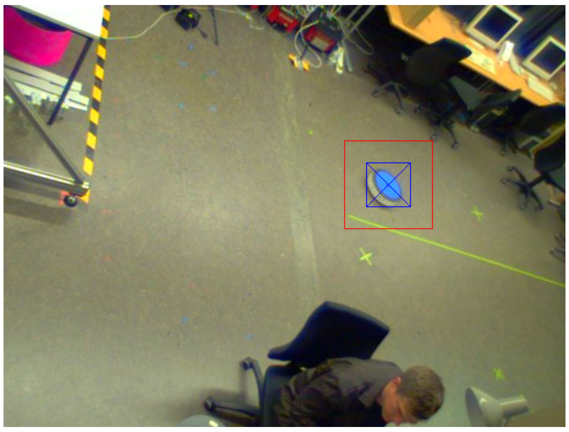
For external data analysis of the performed cleaning runs (section 4.4.1), a custom-built visual tracking system was used (figure 4.15). In order to track the robot, a red disk is mounted on top of the robot (figure 4.11), which is not visible in the robot's panoramic image. The robot's workspace is observed by two cameras mounted statically on the ceiling of the lab. For each camera, the red disk is detected in the camera image and tracked over time using the mean-shift algorithm by Comaniciu and Meer [116]. The marker's center of gravity in the camera image is used to compute an estimate of the robot's world position by direct linear transformation (textbook: [278]). To fuse the estimates of both cameras, a weighted average depending on the agent's distance to the camera is used. By this means, the robot's position but not its orientation can be determined. The resulting tracking accuracy is approximately 1 cm.

For correctly tracking the robot's position, the center of gravity detected by color segmentation

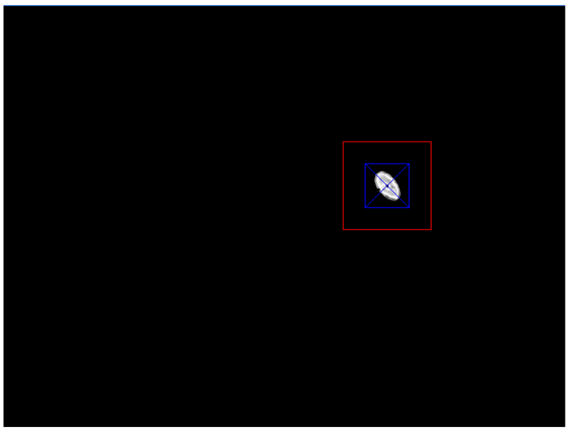


(1) Sketch of the principles

(2) Camera setup. Photo by Lorenz Hillen.



(3) Camera image



(4) Distance image

Figure 4.15.: Passive visual tracking system. Subfigure (1) depicts the principles of the tracking system. Two cameras equipped with wide-angle lenses are mounted statically on the walls of our laboratory observing the experimental area (light gray). The robot's position in the world (black circle) is computed from its position in the two camera images. Subfigure (2) is a photo of the used camera setup (camera: Axis 211 W [D3]; lens: Tamron 13VG308AS Vari-Focal 3–8 mm; mount: custom-built and Manfrotto 484RC2 ball head [I68]). Subfigure (3) shows an example image obtained from one of the cameras. The robot is equipped with a salient tracking target (here, a blue disk and not a red disk as depicted in figure 4.11), which is detected in the camera image by color segmentation. To facilitate color segmentation, the camera's image is oversaturated by setting the saturation parameter to its maximum value. For the pixels inside the search region (red rectangle), the color dissimilarity to the reference color is computed. Subfigure (4) depicts the dissimilarity information with white and black coding identical and dissimilar pixels, respectively. The dissimilarity information is used to track the target, to repeatedly compute its center (blue cross) by applying the mean-shift algorithm (Comaniciu and Meer [116]), and to adaptively adjust the reference color to the current illumination conditions. For estimating the robot's position in the world (black circle in subfigure (1)), the pixel coordinate is converted into a world position by direct linear transformation (textbook: [278]). This requires a calibration of the system which relates a set of known world positions with their corresponding image positions and estimates the mapping between world and image coordinates. As the robot is moving in the plane, distance estimates could in principle be obtained from a single camera image. The tracking accuracy can be increased (i) by calibrating cameras in order to remove lens distortions (textbook: [278]) and (ii) by fusing estimates obtained from two cameras by weighted averaging. As the accuracy of the position estimate decreases with increasing distance to the camera, image regions containing distant areas of the workspace are weighted less reliable than image regions showing nearby regions. With an image resolution of 640 × 480 pixels, the used system is capable of tracking at approximately 5 images per second and has an accuracy of 1 cm. Using a disk as tracking target does not allow to directly compute the robot's orientation. It is rather estimated based on the robot's change of orientation computed from the recorded trajectory data. Similar robot-tracking systems are described in [79, 80, 170, 386, 387]. Figure requires color printing.

of the tracking target has to lie at the center of the target, i.e. at the position where the robot's camera is imaged. In other cases, the estimated position and the true position differ. Such situations occur (i) under certain illumination conditions or (ii) under occlusions of the tracking target. The first aspect includes transitions from sunlit areas to shadow areas within the tracking target. In such cases, the color-segmentation algorithm fails and only detects parts of the target as belonging to the reference color —even though the entire target is visible in the camera image. Thus, the center of gravity of the area belonging to the reference color but not of the entire disk is computed. This restriction required experiments to be conducted under diffuse and constant illumination. The second aspect subsumes occlusions caused by people moving in the lab or by obstacles if the robot moves too close to or underneath them (figure 4.10). In this case, the tracking target is only partially visible for the tracking system, and the center of gravity computed from the visible part is likely to differ from that of the entire target. Therefore, experiments had to be conducted in a static environment and at the center of the lab's free space.

4.4.6. Data Evaluation

The trajectories recorded with the tracking system were analyzed qualitatively and quantitatively. For qualitative evaluation, the resulting trajectories were plotted (figure 4.16). For quantitative data analysis, piecewise polynomials of third order (Matlab function `spline`) were fitted to the recorded robot positions along a lane in order to interpolate between snapshot positions and to estimate the robot's orientation, which cannot be measured by our visual tracking system. Based on this, we computed measures for the inter-lane distances and the resulting cleaning performance. These measures will be described in the following.

4.4.6.1. Inter-Lane Distances

As the proposed trajectory controller (section 4.3.4) is supposed to keep the robot at the desired lane distance Δl from the previous lane, we computed for every snapshot position the robot's true distance to the previous lane. For this purpose, we minimized the spatial distance between the considered snapshot position along the current lane and an arbitrary point along the previous lane approximated by piecewise polynomials. Thus, the search was not restricted to discrete positions of snapshots taken along the previous lane. The resulting inter-lane distances were pooled over all trials of an experiment and analyzed lane-by-lane as well as pooled over all lanes. As performance measures, percentiles and differences between percentiles of the resulting distance distributions were used. Results are shown in figure 4.17 and described in section 4.5.2.

4.4.6.2. Cleaning Performance

To assess the cleaning performance of the proposed navigation strategy, we analyzed the area covered by a simulated suction unit sized 30 cm × 5 cm, which was moved in small steps along the robot's trajectory. By this means, the area A_1 covered exactly once, the overlap between consecutive lanes A_2, and the uncovered area A_0 were computed. The percentages are measured w.r.t. the area covered by an ideal cleaning run (figure 4.10, light-gray areas). Hence, areas outside the ideal area are not considered for the evaluation. Lane changes are excluded from the evaluation assuming the robot's cleaning unit to be switched off. In future work, these measures will be computed w.r.t. the area accessible for the robot. All percentages of A_0, A_1, and A_2 sum up to 100 %; a perfect trial would yield $A_1 = 100.0\,\%$ and $A_0 = A_2 = 0.0\,\%$. Similar measures were also used in related work by Palleja et al. [501], by Rhim et al. [535], and in our recent publication Möller et al. [457].

4.5. Results

The results obtained by experiments 1 and 2 were analyzed qualitatively (section 4.5.1) and quantitatively (sections 4.5.2 and 4.5.3). Thereafter, the results will be discussed in section 4.6.

4.5.1. Qualitative Analysis

Figure 4.16 visualizes the trajectories obtained for experiments 1 and 2, respectively. The figures 4.16.1 and 4.16.2 show all ten trials in one plot in order to analyze the method's repeatability. The other plots show a single trajectory with systematic error to the left ($k = -0.05$; figures 4.16.3 and 4.16.4) and to the right ($k = 0.05$; figures 4.16.5 and 4.16.6). The video [S3] shows a similar experiment except for the number of lanes and the lane length which both had to be reduced in order to fit the camera's field of view.

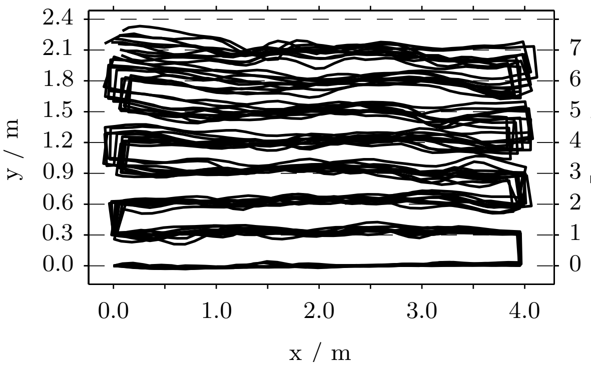
Regarding the repeatability, the trajectories of both experiments are close together with only little variability of up to lane index $i = 3$. For $i > 3$, the repeatability of the lanes decreases especially at the beginning and the end of the lanes. In the middle of each lane (experiment 1: $1\,\mathrm{m} < x < 3\,\mathrm{m}$, experiment 2: $0.5\,\mathrm{m} < x < 1.5\,\mathrm{m}$), the lanes are still close together. Since the robot does not move straight and parallel to the x-axis, but rather oscillates around its desired lane, its orientation at the end of the lane is not parallel to the x-axis. After changing lanes, this deviation can be increased due to inaccuracies during the robot's rotation. At the beginning, the robot has to compensate for the deviation in order to return to a course parallel to the previous lane. At the end of the lane, the robot follows the deviations of the previous lane. Up to $i = 5$ and $i = 8$ for experiments 1 and 2, respectively, the trajectories of different lanes are clearly separable from each other; for larger i, trajectories of different lanes overlap. Thus, the repeatability decreases with increasing number of lanes i.

Analyzing single trajectories reveals that the robot moves on slightly curved lanes which are locally parallel to each other. As the robot keeps the distance to its previous lane constant, it follows the deviations which occurred along the previous lane. One would expect that this leads to controller errors accumulating over time because additional controller errors occur while moving parallel to the previous lane. However, the obtained trajectories do not reveal such controller errors accumulating from lane to lane. Only the last lane of each experiment is usually more curved than the other lanes. During all experiments, we have never observed that the current lane touches or even crosses the previous lane.

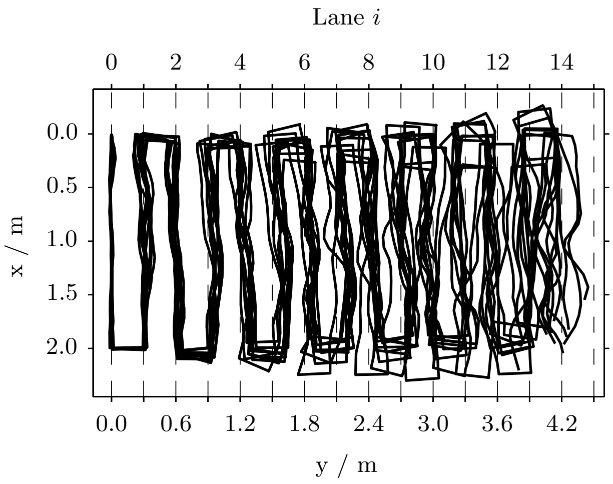
4.5.2. Inter-Lane Distances

The results for analyzing the robot's distance from the previous lane are summarized both in figure 4.17 and in table A.1. For each lane i, the boxes mark the ranges $I^{(i)}_{0.50}$ from the lower quartile $P^{(i)}_{0.25}$ to the upper quartile $P^{(i)}_{0.75}$ containing 50 % of the obtained distance values. The median inter-lane distance $P^{(i)}_{0.50}$ for each lane is depicted by the short horizontal lines dividing the boxes. The whiskers span 90 % of the distance values ranging from $P^{(i)}_{0.05}$ to $P^{(i)}_{0.95}$. We refer to this interval as $I^{(i)}_{0.90}$. In the figure's background, the median inter-lane distances $\bar{P}_{0.50}$ for pooling over all lanes is visualized by the horizontal line. The gray area marks the range $\bar{I}_{0.50}$ between the lower quartile $\bar{P}_{0.25}$ and the upper quartile $\bar{P}_{0.75}$ for pooling over all lanes.

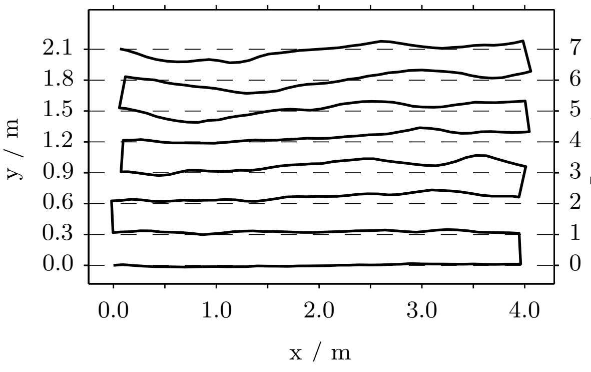
For both experiments, the median lane distance for pooling over all lanes and trials is $\bar{P}_{0.50} = 29.8\,\mathrm{cm}$. The lower and upper quartiles are $\bar{P}_{0.25} = 26.3\,\mathrm{cm}$ and $\bar{P}_{0.75} = 32.4\,\mathrm{cm}$ for experiment 1 (figure 4.17.1), and $\bar{P}_{0.25} = 26.2\,\mathrm{cm}$ and $\bar{P}_{0.75} = 32.2\,\mathrm{cm}$ for experiment 2 (figure 4.17.2). Thus, both experiments achieve almost identical results. With few exceptions, the median values and inter-quartile distances obtained for analyzing single lanes are similar to the overall values. Furthermore,



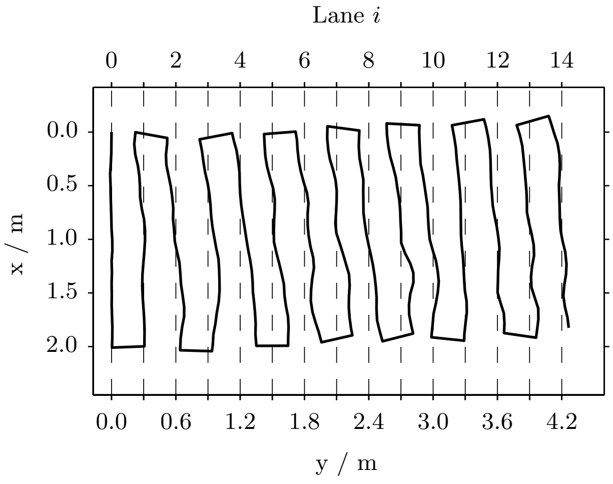
(1) Experiment 1 (pooled data)



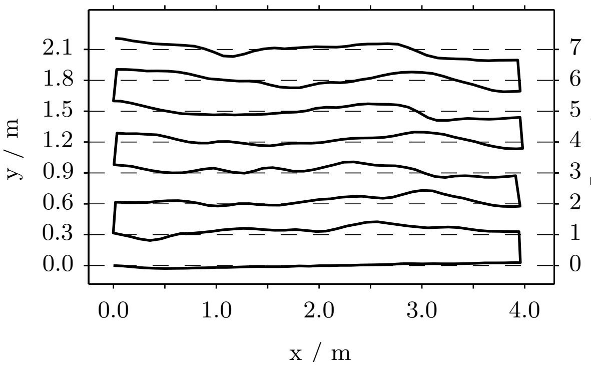
(2) Experiment 2 (pooled data).



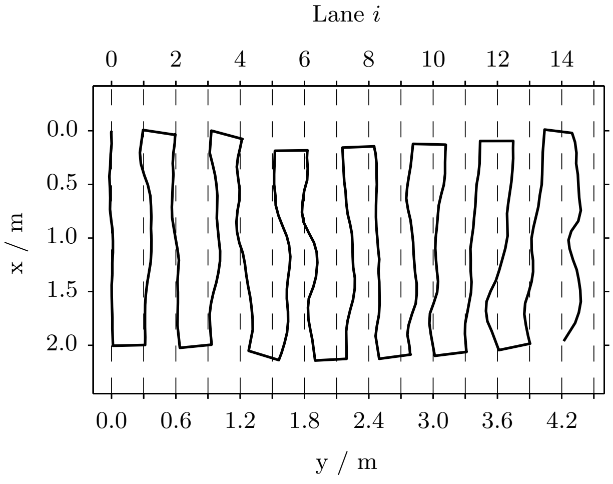
(3) Experiment 1 (systematic error to the left)



(4) Experiment 2 (systematic error to the left)

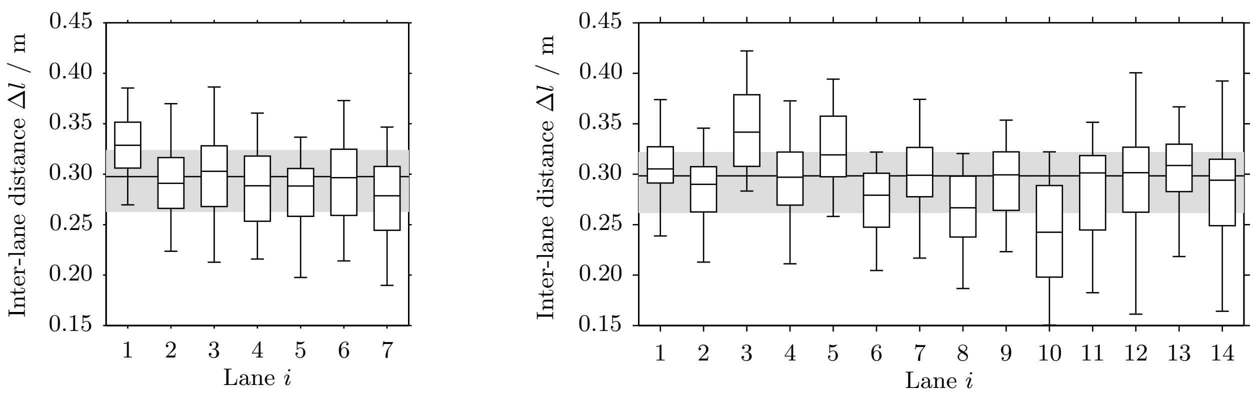


(5) Experiment 1 (systematic error to the right)



(6) Experiment 2 (systematic error to the right)

Figure 4.16.: Cleaning trajectories of experiments 1 and 2. The plots in the left and right column depict the trajectories of experiments 1 and 2, respectively. The top row shows the trajectories pooled over all 10 trials. The second and third row contain a single trajectory obtained with systematic error to the left and the right, respectively.



(1) Experiment 1 **(2)** Experiment 2

Figure 4.17.: Analysis of inter-lane distances of experiments 1 (subfigure (1)) and 2 (subfigure (2)). The boxes visualize for each lane i the range $I_{0.50}^{(i)}$ containing 50 % of the obtained distance values; the median $P_{0.50}^{(i)}$ is depicted by the small horizontal line dividing each box. The whiskers span 90 % of the inter-lane distance values ranging from the $P_{0.05}^{(i)}$ to the $P_{0.95}^{(i)}$ percentiles. The gray area in the figure's background marks the range $\bar{I}_{0.50}$ of inter-lane distances containing 50 % of the values for pooling over all lanes; the overall median $\bar{P}_{0.50}$ is depicted by the horizontal line.

the obtained results seem to be independent of the lane counter i (e.g. they do not increase with increasing lane counter).

For experiment 1 (figure 4.17.1), the minimum and maximum median inter-lane distances are $P_{0.50}^{(7)} = 27.9\,\text{cm}$ and $P_{0.50}^{(1)} = 32.9\,\text{cm}$, respectively. The boxes span inter-quartile distances ranging from $I_{0.50}^{(1)} = 4.5\,\text{cm}$ to $I_{0.50}^{(6)} = 6.5\,\text{cm}$; the whiskers span inter-percentile distances between $I_{0.90}^{(1)} = 11.5\,\text{cm}$ and $I_{0.90}^{(3)} = 17.3\,\text{cm}$. For both measures, the range from the lower percentile to the median is larger than the range from the median to the upper percentile. The measures for even lanes (robot is moving in positive x-direction of $\langle W \rangle$, $i = 2, 4, \ldots$) and for odd lanes (moving in negative x-direction, $i = 1, 3, \ldots$) do not differ.

The median inter-lane distances obtained for the 14 lanes of experiment 2 (figure 4.17.2) vary between $P_{0.50}^{(10)} = 24.2\,\text{cm}$ and $P_{0.50}^{(3)} = 34.1\,\text{cm}$. For the first 11 lanes, the median inter-lane distances of the even and odd lanes differ by several centimeters with the values of the odd lanes being larger. Most inter-quartile distances $I_{0.50}$ are approximately 6 cm, with $I_{0.50}^{(1)} = 3.6\,\text{cm}$ and $I_{0.50}^{(10)} = 9.1\,\text{cm}$ being the minimum and maximum inter-quartile distances. Most values of $I_{0.90}$ are approximately 15 cm; minimum and maximum are $I_{0.90}^{(12)} = 11.7\,\text{cm}$ and $I_{0.90}^{(6)} = 23.9\,\text{cm}$, respectively.

4.5.3. Cleaning Performance

For analyzing the cleaning performance, we computed for each of the 20 trials the percentages of the uncovered area A_0, of the area A_1 covered exactly once, and of the area A_2 covered exactly twice due to overlap between consecutive lanes[1]. As reference area, we used the area covered by ideal cleaning runs as depicted in figure 4.10; parts of the robot's trajectory outside these areas have not been considered. The complete data is given in table A.2 and figures A.1 to A.4.

For experiment 1, the performance varies between $A_0 = 9.4\,\%$, $A_1 = 81.6\,\%$, and $A_2 = 9.0\,\%$ for the worst and $A_0 = 6.4\,\%$, $A_1 = 87.7\,\%$, and $A_2 = 5.9\,\%$ for the best trial with an average performance of $\bar{A}_0 = 7.9\,\%$, $\bar{A}_1 = 85.4\,\%$, and $\bar{A}_2 = 6.7\,\%$. The average performance of experiment 2 is slightly worse: we obtained a performance of $\bar{A}_0 = 10.3\,\%$, $\bar{A}_1 = 81.5\,\%$, and $\bar{A}_2 = 8.2\,\%$. This experiment also exhibits a larger variability between trials because the performance ranges from $A_0 = 17.2\,\%$, $A_1 = 71.7\,\%$, and $A_2 = 11.1\,\%$ to $A_0 = 5.9\,\%$, $A_1 = 89.3\,\%$, and $A_2 = 4.8\,\%$. Figure 4.18 shows for

[1]In more general experiments one would rather measure the area covered *more than once.*

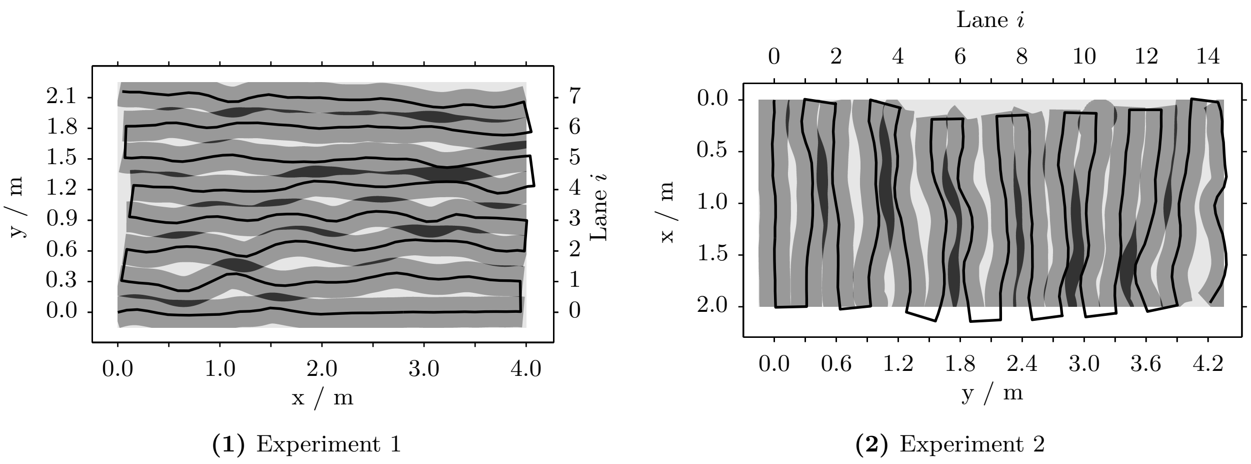


Figure 4.18.: Analysis of the cleaning performance for experiments 1 (subfigure (1)) and 2 (subfigure (2)). For each experiment, one trial with a performance close to the experiment's average performance is shown. The graphs depict the robot's trajectory (black line) together with the uncovered area A_0 (light gray), the area A_1 covered once (gray) and the area A_2 covered twice (dark gray). All trials of both experiments are depicted in figures A.1 to A.4.

each experiment the trajectory achieving the performance which is most similar to the average performance. The figure reveals that the uncovered area is mostly due to gaps between consecutive lanes because of the robot's distance to the previous lane being too large. Both the uncleaned area and the overlap between lanes do not increase with increasing lane number i. Rather, the method seems to achieve equal performance over all lanes.

4.6. Discussion and Conclusions

The results presented in sections 4.5.1 to 4.5.3 show that the proposed trajectory controller is capable of guiding the robot along parallel lanes while achieving a good coverage of the robot's workspace. Although several of the assumptions made for the controller's derivation (equations (4.4) and (4.19) to (4.21)) are not completely satisfied, the obtained results reveal that the assumptions were chosen reasonably and that the controller exhibits a certain robustness against their violations. In the remainder of this section, aspects related to *partial pose estimation*, *dense topo-metric maps*, *image disturbances*, and *further cleaning strategies* will be discussed in more detail (sections 4.6.1 to 4.6.4); the conclusion will be summarized in section 4.6.5.

4.6.1. Partial Pose Estimation

Keeping the robot's distance to its previous lane constant means that the robot's desired trajectory directly depends on its previous lane. This approach is potentially prone to controller errors accumulating from lane to lane: in case the robot does not exactly follow its current desired trajectory but oscillates around it, new controller errors occur, and the resulting trajectory will influence the desired trajectory of the following lane. Nevertheless, such accumulating errors did not occur during our experiments (figures 4.16 to 4.18). We think that this is due to the following two reasons: (i) the selection of different triangulation sets and the fusion of the estimates provided by each of these sets and (ii) the moving target distance t (equation (4.26)). Both aspects seem to have a dampening influence on the robot's trajectory causing the robot to follow the previous lane smoothly. From these results we conclude that our approach relying on partial pose estimation is sufficient to guide the robot along parallel and meandering lanes.

In case accumulating controller errors should occur with the proposed method (e.g. under different environmental conditions), they could be reduced (i) by correcting the estimates attached to the corresponding place nodes at a later point in time, (ii) by selecting neighbors not along the previous lane but along its prepredecessor, or (iii) by making the desired lane independent of the previous lane. Subsequent improvements of initial estimates are usually performed by SLAM methods (sections 2.2.2 and 3.6.1.2) and are essential for achieving globally consistent maps. However, they are also the computationally most demanding parts of SLAM algorithms. For our future work, we hope to avoid this step by completely covering the robot's workspace by several locally consistent cleaning segments rather than building a single globally consistent map (section 7.3.3). Selecting snapshots stored along the last but one lane could probably improve the method's performance. However, it would also require a special treatment of the second cleaning lane ($i = 1$) and is for this reason not considered.

Making the desired lane independent of the previous lane requires to specify the robot's desired lane with respect to world coordinates. However, this would require to estimate the robot's full pose $\hat{\boldsymbol{s}}^{(i,j)} = (\hat{x}^{(i,j)}, \hat{y}^{(i,j)}, \hat{\theta}^{(i,j)})^{\top}$, which could be determined by intersecting the rays defined by the triangle's base points $\boldsymbol{s}^{(i-1,j')}$ and $\boldsymbol{s}^{(i-1,j'')}$ and by the corresponding homing angles α_{k_1} and α_{k_2} (figure 4.6). The computations required to estimate the robot's full pose use the same orientation and bearing estimates than the proposed method. We are therefore of the opinion that *directly* using the full pose estimates computed by this means would not considerably improve the navigation accuracy of our method. We only expect improvements for *fusing several estimates* by a Bayesian filtering framework (such as the Kalman or the particle filter; textbooks: [586, 630] and section 3.3.2.1) which could use the sketched approach for its update step.

In parallel to the work described here, we implemented two similar control strategies based on the *Kalman filter* and on the *particle filter*. The Kalman-filter method was initially implemented in the diploma thesis by Janina de Jong [314] (co-supervised by Prof. Dr. Ralf Möller and Lorenz Hillen) and later on refined in the course of her PhD project. Since the parameter tuning of the Kalman-filter method has proven to be subtle, Prof. Dr. Ralf Möller implemented a similar method relying on a particle-filter framework for position estimation (Möller et al. [457]). The parameters of this method can be easily tuned in order to achieve good results. Prof. Dr. Ralf Möller recently compared the method proposed in this chapter and the particle-filter method. The comparison revealed that the particle-filter method produces straighter lanes, but both methods achieve a comparable level of parallelism [457]. The Kalman-filter method and the particle-filter method both compute the robot's full pose and therefore allow to specify the desired lane in world coordinates. For following the trajectory standard methods for trajectory control can be applied (review: [462]; textbook: [41, 586]). However, even such frameworks cannot completely prevent error accumulation from lane to lane. Rather, a drift of the position estimate is likely to occur for any visual odometry algorithm (reviews: [191, 563]). This problem is inherent to all methods using former robot positions as landmarks because the correction step of the filter would still refer to snapshot images stored along the previous lane. Thus, position errors and the corresponding uncertainties on the previous lane could affect the estimates on the current lane. Although being computationally more complex, we prefer the trajectory controller based on the particle filter for our future work because (i) it produces straighter cleaning lanes and (ii) it allows for specifying trajectories in world coordinates and applying standard controllers for trajectory following. The latter aspect can facilitate further strategies required for completely covering complex-shaped workspaces (section 7.3.2) because such strategies require path following in order to approach uncleaned areas or the robot's charging station.

4.6.2. Dense Topo-Metric Maps

The results have also shown that dense topo-metric maps are applicable for completely covering segments of the robot's entire workspace. By adding new snapshots at regular distances, a sufficiently large number of snapshots is available for correcting the robot's position while still preserving the advantages of topological maps (section 4.2). We consider the choice of the inter-snapshot distance $\Delta s = 10\,\text{cm}$ to be sufficient for our application and do not expect that smaller choices of Δs could improve the performance of the proposed controller. This is due to the observation (unpublished data) that home vectors computed from a current view and a series of neighboring snapshots taken along the previous lane have correlated errors with the strength of the correlation depending on the spatial distance between the snapshots. Furthermore, smaller choices of Δs would also increase the memory requirements of the resulting topo-metric map. We expect that more complex cleaning strategies for completely covering complex-shaped workspaces can be developed based on dense topo-metric maps (section 4.6.4).

4.6.3. Influence of Image Disturbances

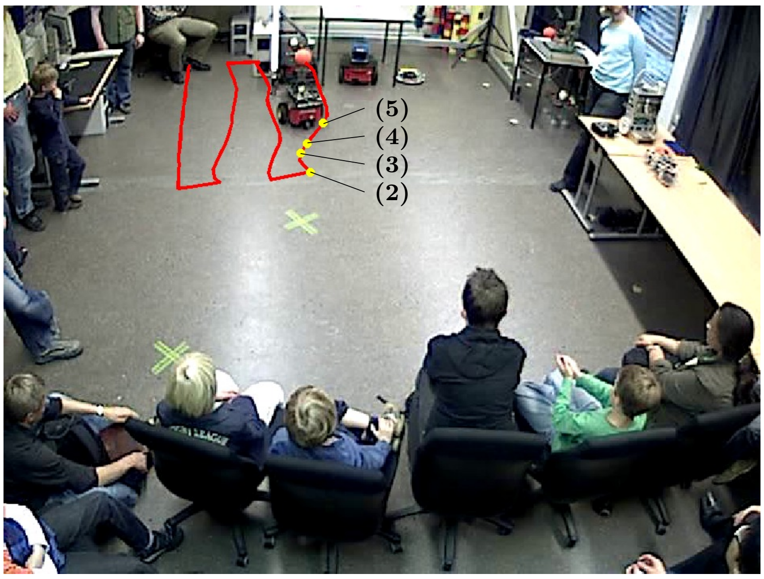
Like any appearance-based method, local visual homing and hence also the proposed trajectory controller are prone to image disturbances as, among others, caused by dynamic scene changes (section 3.2.3.2), changes of the illumination (section 3.2.3.2), or a tilt of the robot if rolling over a cable or a carpet border. In its current implementation, the controller is not capable of detecting or reacting to such situations. A certain, but probably limited, robustness should arise from fusing several estimates as described in section 4.3.3.4. By this means, the proposed strategy would be converted from a *deterministic* approach to a *probabilistic* navigation method. If the underlying homing method is not robust against such image disturbances, the computed home vectors will be erroneous and the estimated inter-lane distance will deviate. We are of the opinion that achieving robustness against image disturbances should be part of the underlying homing method and not of the control algorithm. This assures that correct home vectors are passed to the controller.

As shown by Möller, Krzykawski, and Gerstmayr [451] and observed in many unpublished real-robot experiments, the min-warping method (section 4.4.2) used in this chapter is capable of computing accurate home vectors over a wide range of different environments. Although we have not systematically investigated its robustness against image disturbances, we observed a certain tolerance against illumination changes and dynamic scene changes. Figure 4.19 gives an impression how the method performs under strong disturbances due to dynamic changes of the scene. Even though the method will not be fully invariant against image disturbances like changing illumination conditions or dynamic scene changes, we expect it to at least tolerate a certain amount of image disturbances. In case a sufficient robustness against image disturbances cannot be achieved by local visual homing methods alone, further strategies for detecting image disturbances could consult the dense topo-metric map. Section 7.3.1 describes such approaches in the context of detecting dynamic scene changes.

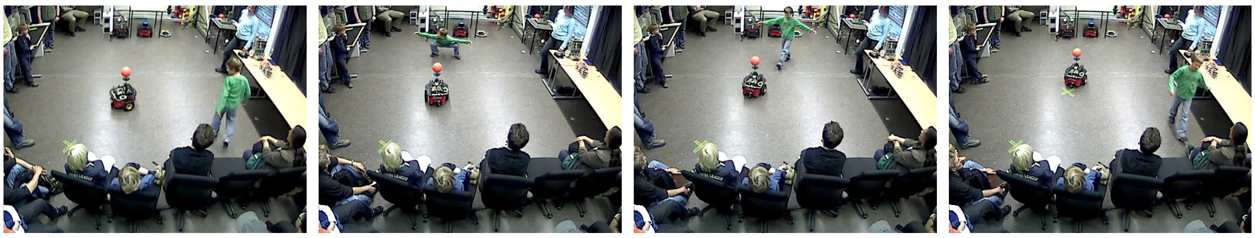
Further aspects which can cause homing to fail are featureless environments or anisotropic distributions of visible objects. In the former case, our warping method (section 4.4.2) could not establish matches between image columns, whereas the second case would result in erroneous home-vector estimates [190, 379, 452]. For a potential product, such special cases should be detected and backup strategies such as random walk should be provided by the robot's control scheme (section 7.3.2).

4.6.4. Implications for Further Cleaning Strategies

In case only a single, rectangular segment is cleaned, the uncleaned area is due to gaps between consecutive cleaning lanes. Thus, the covered area could be further increased by reducing the desired inter-lane distance Δl. However, this would be at the cost of increasing the proportion of repeated coverage. The results presented in section 4.5 show that relatively large areas can be covered by the proposed method. As real apartments are usually more cluttered, we expect the maximum



(1) Overview and trajectory



(2) **(3)** **(4)** **(5)**

Figure 4.19.: Trajectory controller and dynamic scene changes. The figure shows five images from a video sequence recorded during a demonstration of the proposed method given at the science festival GENIALE [I14] (Bielefeld, October 2008). We demonstrated an earlier implementation of the proposed trajectory controller running on an Adept MobileRobots Pioneer 3DX [D2] mobile robot equipped with a hyperbolic mirror (Accowle Large Type Wide Angle [D1]; figure 3.7.3). Subfigure (1) depicts the robot's trajectory at the end of the fourth cleaning lane. The trajectory (red line) was recovered manually by analyzing the video sequence. While the robot was moving along the fourth lane, a boy was moving close to the robot trying to disturb it. The resulting lane is more curved than the previous ones, but the method did not completely fail. Along the fourth lane, four points in time are marked by yellow circles. The circles mark the situations depicted in subfigures (2) to (5). Figure requires color printing.

area which can be covered by a single cleaning segment to be much smaller (figure 7.3). For this reason, good strategies for combining several cleaning segments and for keeping the first lane of these segments straight have to be developed. Based on these strategies, we hope to be able to circumvent the computationally demanding steps of subsequent position corrections by building a hierarchical representation of space with a purely topological representation on top of several locally consistent submaps each corresponding to a single cleaning segment (section 7.3.3). Even if real apartments require the decomposition into several smaller cleaning segments, we think that a decomposition into rectangular segments of parallel lanes is the best choice for most of the cases. For special cases which cannot be covered efficiently by parallel and meandering lanes, alternative cleaning strategies or backup strategies could be provided by a more elaborated framework of cleaning strategies (section 7.3.2).

4.6.5. Conclusions

We conclude that the proposed method is capable of guiding the robot along parallel and meandering lanes and of achieving a good coverage with only a small portion of uncleaned areas or repeated coverage between lanes. It is therefore suitable for covering a single segment of the robot's workspace and concurrently mapping it. Such segments can then be combined by a more sophisticated control architecture to systematically and completely cover complex-shaped workspaces (section 7.3.2). We furthermore conclude that the dense topo-metric map built while the robot cleans its current segment is an appropriate spatial representation for cleaning-robot control. With their fine spatial resolution, these maps are well suited for frequently repeating the distance-estimation steps described in section 4.3 and for achieving precise navigation. For future work, we will abandon the concept of partial pose estimation propagated in this chapter in favor of our particle-filter method. The particle-filter method yields straighter lanes and knowing the robot's full pose can facilitate more elaborated cleaning structures. These two advantages outweigh the larger computational effort of methods estimating the robot's full pose with a Bayesian filter. To this end, a cleaning-trajectory controller based on partial pose estimation has to be considered a theoretically intersecting concept but —at least for the proposed algorithm— has limitations for the application in combination with full-fledged cleaning strategies.

4.7. Future Working Directions

Several future working discussions were already mentioned in sections 4.6.1 to 4.6.4. They mainly include working towards a complete framework of cleaning strategies (section 7.3.2). Such a framework is required to completely cover complex-shaped workspaces by combining several segments of parallel and meandering lanes as resulting from the proposed trajectory controller. The resulting system will make use of the underlying dense topo-metric map for detecting areas which still need to be cleaned, for loop-closure detection to avoid repeated coverage (e.g. by applying the methods proposed in chapters 5 and 6), for planning and approaching new lanes or new cleaning segments, and for planning and following paths back to the robot's charging station. Based on the decomposition of the robot's workspace into several segments of meandering lanes, interesting possibilities arise in the context of hierarchical mapping. These will be further described in section 7.3.3. The dense topo-metric map built by the trajectory controller proposed in this method could also be used to increase the robustness of the navigation strategies against dynamic scene changes. For details please refer to section 7.3.1.

5. Holistic Loop-Closure Detection and Visual Compass

We present loop-closure detection methods relying on pixel-by-pixel comparisons of images and on a visual compass. As the proposed methods compare entire images, we refer to them as holistic methods. In order to increase the robustness against changes of the illumination, we apply image preprocessing methods prior to image comparison.
The chapter is structured as follows: section 5.1 briefly introduces loop-closure detection based on global image comparisons. A detailed description of the used methods follows in section 5.2. The experimental procedure and evaluation methods are explained in section 5.3. Sections 5.4 and 5.5 describe the experiments for the standard method and an accelerated variant, respectively. The chapter ends with a final summary and an outlook to future work (section 5.6).
Parts of this chapter are inspired by the bachelor's thesis of Björn Böttcher [60] supervised by Lorenz Hillen and Dr. Wolfgang Stürzl. For the computation of AUC values (section 5.3.1), a program written by Oliver Schlüter during the course of his bachelor's thesis [575] (supervised by Lorenz Hillen and Martin Krzykawski) is used; the other evaluation and visualization software was implemented by Lorenz Hillen. Section 5.3.1.1 of this thesis is an extension of section IV of our conference publication GERSTMAYR-HILLEN *et al. [223].*
Preliminary results from this chapter have already been presented as a poster GERSTMAYR-HILLEN *and* MÖLLER *[220].*

5.1. Introduction

In this chapter, we propose holistic loop-closure detection methods relying on pixel-by-pixel comparisons between the robot's current camera image and images stored in the dense topo-metric map (sections 3.6.3.2 and 4.2) of the robot's workspace. By such comparisons, an image dissimilarity value is computed, which can then be used to detect loop closures by classifying whether or not the compared images are identical. For comparing images, we apply global image dissimilarity functions which —in contrast to local image dissimilarity functions— do not subdivide the images into local patches (GIACHETTI [225]). In the following, we prefer the term "image dissimilarity" over the commonly used term "image distance" in order to emphasize the difference between image dissimilarity and the spatial distance between the positions of image acquisition. Loop-closure detection by global image comparisons requires that the compared images are aligned w.r.t. a common reference direction. We therefore apply the holistic visual compass method (section 3.4.2.1) proposed by ZEIL, HOFFMANN, and CHAHL [718] to align the images prior to loop-closure detection. It aligns the images by shifting one of the images step-by-step and minimizing the global image dissimilarity between the shifted image and the reference image. Thus, the residual of this optimization can be directly used for loop-closure detection. In order to achieve the best possible loop-closure detection performance, we test the visual compass with a wide range of different image dissimilarity functions. As most of the tested dissimilarity functions are not robust against changes of the illumination, we apply image preprocessing methods such as edge detection or early-vision models before comparing the images. By this means, we expect to achieve invariance or at least tolerance against changes of the illumination.

Analogous to *holistic representations of places* (section 3.3.1.1), using the entire image to represent places based on visual information, we refer to the methods proposed in this chapter as *holistic loop-closure detection*. In comparison to loop-closure detection based on global image signatures (chapter 6), the methods proposed in this chapter are computationally more complex because images are compared pixel-by-pixel and not by matching lower-dimensional signatures. The methods tested in this chapter solve the correspondence problem implicitly by aligning the images w.r.t. a common reference direction. This distinguishes pixel-based methods from feature-based approaches to loop-closure detection. Such methods are currently the standard approach to loop-closure detection. They extract features by a point-of-interest detector (section 3.3.1.3), characterize these features by a feature descriptor and establish correspondences between such descriptors, frequently in conjunction with an outlier removal procedure. The advantages of these methods are (i) that they do not require the images to be aligned w.r.t. a common reference direction for establishing correspondences and (ii) that —due to using local image patches for all involved operations— they could exhibit a better robustness against illumination changes. However, they require a large algorithmic effort and in some cases also a training stage to allow for efficient image comparisons (section 3.3.1.3).

As already outlined in section 3.2.3.3, not correctly detecting loop closures causes maps to become inconsistent and navigation to fail. For an autonomous cleaning robot, this results in uncovered areas or in repeated coverage caused by gaps or overlap between neighboring cleaning segments. Furthermore, if several cleaning segments are combined to completely cover the robot's workspace (section 7.3.2), loop-closure detection can be used to find shortcuts between segments. Since the robot's position estimate is likely to drift over time, loop-closure algorithms need to be purely vision-based and cannot rely on the position estimate. For these reasons, visual loop-closure detection is an essential subsystem for such a robot. The following section describes the proposed holistic loop-closure detection methods in more detail.

5.2. Methods

The algorithm for loop-closure detection based on global image comparisons is sketched in figure 5.1: the robot's current camera image $\boldsymbol{C}_P$ is unfolded to a cylindrical camera image $\boldsymbol{I}_P$. If the robot has traveled a predefined distance since adding the last place node, a new node is added to the map which has the image $\boldsymbol{I}_P$ and an estimate of the robot's current position attached (see chapter 4 on details of the map-building and position estimation). As most of the image dissimilarity functions used for image comparisons are not robust against changes of the illumination, the images are preprocessed prior to image comparison by applying an appropriate preprocessing function p. This step is supposed to reduce or even eliminate the influence of illumination changes. Preprocessing yields the preprocessed image $\boldsymbol{P} = \mathrm{p}(\boldsymbol{I}_P)$. If storage capacity is not a limiting factor or repeated applications of the preprocessing function should be avoided, the preprocessed image $\boldsymbol{P}$ can also be attached to the corresponding place node. For loop-closure detection, the preprocessed image $\boldsymbol{P}$ is compared to preprocessed images $\boldsymbol{Q}_i = \mathrm{p}(\boldsymbol{I}_{Q_i})$ stored in the topological map. For each considered image pair, this results in the image dissimilarity $\hat{\ell}_i$ which is used to classify whether the images $\boldsymbol{I}_P$ and $\boldsymbol{I}_{Q_i}$ are identical or not by a binary classification of $\hat{\ell}_i$ w.r.t. the decision threshold ℓ_t. Depending on the number of images stored in the map, this procedure requires a large number of image comparisons. By selecting the images $\boldsymbol{Q}_j$ only from the borders of cleaning segments, the number of comparisons can be considerably reduced. Nevertheless, efficient image comparisons are required to make the proposed methods applicable on a real cleaning robot. For sake of simplicity, we will in the following omit the subscript i and refer to the second image solely as $\boldsymbol{Q}$.

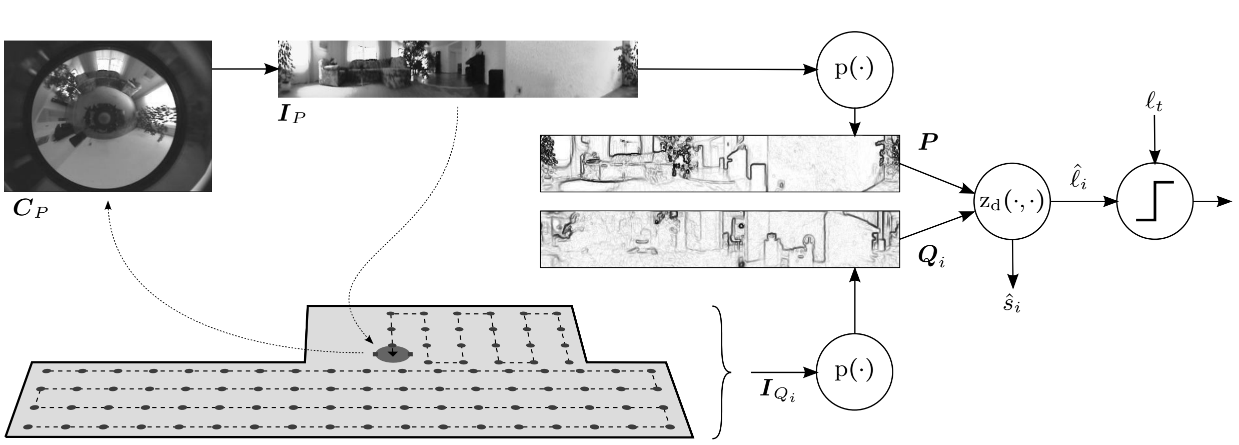


Figure 5.1.: Principles of holistic loop-closure detection. By covering the robot's workspace with segments of parallel lanes (dashed lines), a dense topo-metric map (sections 3.6.3.2 and 4.2) is built (edges not shown). Loop closures are only tested between the current image and the border of cleaning segments to avoid that the robot enters an already cleaned area. For loop-closure detection, the images $\boldsymbol{I}_P$ and $\boldsymbol{I}_{Q_i}$ are preprocessed by a preprocessing function p. The preprocessed images $\boldsymbol{P}$ and $\boldsymbol{Q}_i$ are compared by the compass function $\mathrm{z_d}$, which aligns the images w.r.t. a common reference direction and computes the dissimilarity $\hat{\ell}_i$ of the considered image pair (Zeil, Hoffmann, and Chahl [718]). The images can be aligned by shifting one of the images by $\hat{s}_i$ columns; the *discrete* compass shift $\hat{s}_i$ could also be expressed as angle $\hat{\psi}_i$. The dissimilarity value $\hat{\ell}_i$ is used for classification w.r.t. a classification threshold ℓ_t.

As image dissimilarity functions (reviews: [19, 91, 92, 225, 549, 635]) are not invariant under rotations of the robot, computing the image dissimilarity $\hat{\ell}$ requires the images $\boldsymbol{P}$ and $\boldsymbol{Q}$ to be aligned w.r.t. a common reference direction. This can be achieved by applying a visual compass method (section 3.4). As the holistic compass methods implicitly involve pixel-by-pixel comparisons of the compared images, these methods are best suited for loop-closure detection based on global image comparisons. The method proposed by Zeil, Hoffmann, and Chahl [718] iteratively shifts the panoramic image $\boldsymbol{Q}$ by s columns, resulting in the shifted image $\boldsymbol{Q}^{(s)}$. In each step, the image dissimilarity

$$\ell(s) = \mathrm{d}\left(\boldsymbol{P}, \boldsymbol{Q}^{(s)}\right) \tag{5.1}$$

between $\boldsymbol{P}$ and the shifted image $\boldsymbol{Q}^{(s)}$ is computed (figure 5.2). By minimizing $\ell(s)$ over all possible column shifts s ($0 \le s < w$ with w being the image width), the compass function

$$\mathrm{z_d}(\boldsymbol{P}, \boldsymbol{Q}) = \min_{s=0}^{w-1} \ell(s) \tag{5.2}$$

$$= \hat{\ell} \tag{5.3}$$

computes the image dissimilarity $\hat{\ell}$ between images $\boldsymbol{P}$ and $\boldsymbol{Q}$ of width w. The compass shift is obtained by computing the shift

$$\hat{s} = \arg\min_{s=0}^{w-1} \ell(s) \tag{5.4}$$

which resulted in $\hat{\ell}$. Throughout this chapter we consider the compass shift to be *discrete* and therefore stick to $\hat{s}$ rather than denoting it by the angle ψ as is the case in chapters 3 and 4. The compass method's overall complexity is $\mathcal{O}(w^2h)$ because for each of the w possible image shifts wh pixel-by-pixel comparisons have to be computed. In the following, the tested image preprocessing methods (section 5.2.1) and dissimilarity functions (section 5.2.2) will be described. A more detailed analysis of computing times for the proposed methods and tested parameter sets is given in sections 5.4.4 and 5.4.5.

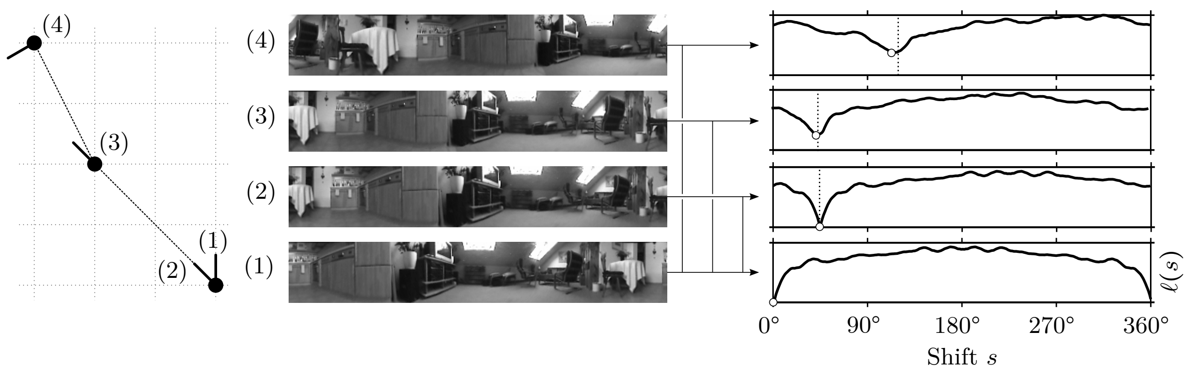


Figure 5.2.: Properties of the vision-based compass proposed by Zeil, Hoffmann, and Chahl [718]. The properties are outlined assuming the following trajectory (left column): the robot is initially located at position (1) facing upwards, rotates (2), translates (3) and finally reaches pose (4). The middle column shows from bottom to top the images acquired at positions (1) to (4). The right column visualizes the dissimilarity $\ell(s)$ vs. the compass shift s for applying the compass method to images (1) and (1) (bottom), (1) and (2), (1) and (3), and (1) and (4) (top). With increasing spatial distance to the reference position (1), the image dissimilarity of the minimum $\ell_{\min}$ (open dot) increases. This effect is exploited for loop-closure detection. The shift $s_{\min}$ leading to $\ell_{\min}$ is used as estimate for the change of the robot's orientation between the compared snapshots. For increasing spatial distance, the estimate deviates stronger from the true change of orientation (vertical dotted lines). As distance function, the sum of squared differences ($\mathrm{d}_{\texttt{ssd}}$) was used; all images were taken from the `roeben` database (section 5.3.1.1).

5.2.1. Image Preprocessing Functions

The image preprocessing functions play an essential role for successful and robust loop-closure detection: they are supposed to reduce the influence of changes of the illumination on image intensities, and they should facilitate the loop-closure detection based on the image dissimilarity computed in the subsequent processing step. For this purpose, an ideal preprocessing method should exhibit robustness against perceptual aliasing[1] (i.e. it should be able to cope with spatially different places having identical visual appearance; section 3.2.3.1) as well as robustness against perceptual variability (i.e. they should tolerate a certain amount of illumination changes and dynamic scene changes; section 3.2.3.2):

$$\mathrm{p}(\boldsymbol{I}(\boldsymbol{x})) = \mathrm{p}(\boldsymbol{I}'(\boldsymbol{x}')) \text{ iff } \boldsymbol{x} = \boldsymbol{x}', \text{ and} \tag{5.5}$$
$$\mathrm{p}(\boldsymbol{I}(\boldsymbol{x},t)) = \mathrm{p}(\boldsymbol{I}(\boldsymbol{x},t')). \tag{5.6}$$

Thus, two preprocessed images $\boldsymbol{P} = \mathrm{p}(\boldsymbol{I}(\boldsymbol{x}))$ and $\boldsymbol{P}' = \mathrm{p}(\boldsymbol{I}'(\boldsymbol{x}'))$ should be identical if and only if the original images $\boldsymbol{I}$ and $\boldsymbol{I}'$ were acquired under identical robot poses $\boldsymbol{x} = (x, y, \theta)^\top$ and $\boldsymbol{x}' = (x', y', \theta')^\top$. Furthermore, preprocessed images are supposed to be identical if they were acquired at a single position in space but at different points t and t' in time. The image preprocessing methods p tested in this chapter can be categorized into the identity function ($\mathrm{p}_{\texttt{id}}$), first-order and second-order edge detectors ($\mathrm{p}_{\texttt{pw}}$, $\mathrm{p}_{\texttt{sob}}$, $\mathrm{p}_{\texttt{lap}}$, and $\mathrm{p}_{\texttt{dog}}$), early-vision models ($\mathrm{p}_{\texttt{dl}}$, $\mathrm{p}_{\texttt{dlc}}$, and $\mathrm{p}_{\texttt{dv}}$), and histogram equalization ($\mathrm{p}_{\texttt{heq}}$). The categorization of preprocessing functions is depicted in figure 5.3, and example images are shown in figure 5.7.

Identity Function

The *identity function*

$$\mathrm{p}_{\texttt{id}}(\boldsymbol{I}) = \boldsymbol{I} \tag{5.7}$$

does not change the intensities of image $\boldsymbol{I}$. It is tested for the sake of completeness and to test whether or not image preprocessing methods can increase the tolerance against illumination changes.

[1]"Perceptual aliasing" is often referred to as "spatial aliasing".

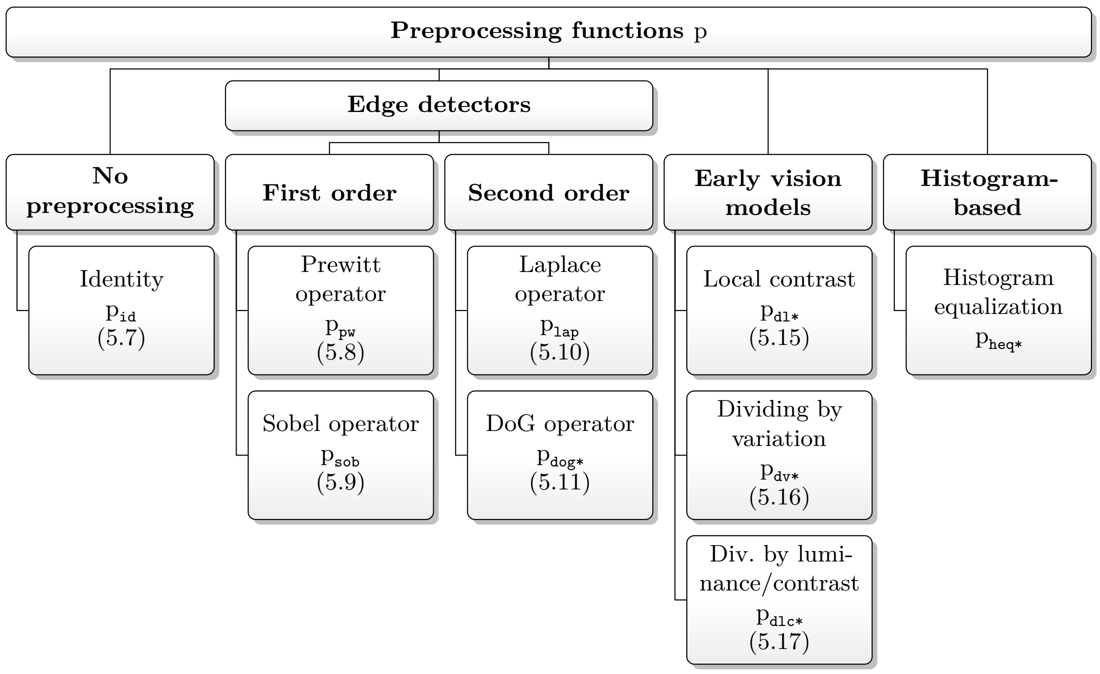

Preprocessing functions p
Edge detectors
No preprocessing
First order
Second order
Early vision models
Histogram-based
Identity p_{id} (5.7)
Prewitt operator p_{pw} (5.8)
Sobel operator p_{sob} (5.9)
Laplace operator p_{lap} (5.10)
DoG operator p_{dog*} (5.11)
Local contrast p_{dl*} (5.15)
Dividing by variation p_{dv*} (5.16)
Div. by luminance/contrast p_{dlc*} (5.17)
Histogram equalization p_{heq*}

Figure 5.3.: Categorization of preprocessing functions used for holistic loop-closure detection. Preprocessing functions tested with several parameters are marked by a star (∗). The numbers given in parentheses refer to the equation defining the corresponding function (except for histogram equalization which is only described in words).

Edge Detectors

Edges are image features which are frequently considered to be invariant against illumination changes (e.g. [644]). This assumption only holds for edges resulting from static objects but is violated for edges resulting from shadow casting because the image location and strength of such edges can change over time due to changes of the illumination conditions. Nevertheless, we test several edge detectors as preprocessing methods. As first-order edge detectors, we use the magnitudes of the intensities of the images filtered with horizontal and vertical *Prewitt* and *Sobel operators*:

$$\mathrm{p}_{\mathtt{pw}}(\boldsymbol{I}) = \sqrt{(\boldsymbol{K}_{\mathtt{pw}} * \boldsymbol{I})^2 + (\boldsymbol{K}_{\mathtt{pw}}^{\top} * \boldsymbol{I})^2} \text{ with } \boldsymbol{K}_{\mathtt{pw}} = \begin{pmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{pmatrix} \text{ and} \tag{5.8}$$

$$\mathrm{p}_{\mathtt{sob}}(\boldsymbol{I}) = \sqrt{(\boldsymbol{K}_{\mathtt{sob}} * \boldsymbol{I})^2 + (\boldsymbol{K}_{\mathtt{sob}}^{\top} * \boldsymbol{I})^2} \text{ with } \boldsymbol{K}_{\mathtt{sob}} = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}. \tag{5.9}$$

As second-order corner detectors, we test the *Laplacian*

$$\mathrm{p}_{\mathtt{lap}}(\boldsymbol{I}) = \boldsymbol{K}_{\mathtt{lap}} * \boldsymbol{I} \text{ with } \boldsymbol{K}_{\mathtt{lap}} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix} \tag{5.10}$$

and the *difference of Gaussians* (DoG) operator:

$$\mathrm{p}_{\mathtt{dog}}(\boldsymbol{I}) = \big(\boldsymbol{K}_{\mathtt{gau}}(\sigma_1) - \boldsymbol{K}_{\mathtt{gau}}(\sigma_2)\big) * \boldsymbol{I}. \tag{5.11}$$

The DoG operator is obtained by subtracting the Gaussian filter kernel $\boldsymbol{K}_{\mathtt{gau}}(\sigma_2)$ with standard deviation σ_2 from the filter kernel $\boldsymbol{K}_{\mathtt{gau}}(\sigma_1)$ with standard deviation σ_1. The filter kernels are computed as a discretization of the 2D Gaussian function

$$\mathrm{gauss}(x, y, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right). \tag{5.12}$$

Early Vision Models

Preprocessing functions based on models of the early visual system of mammals are considered here because this part of the visual system is essential for achieving illumination invariance. These first stages of the visual pathway and preprocess the incoming visual stimuli for further processing in higher cortical brain areas. For further details the reader is referred to the reviews focusing on modeling aspects (reviews: [81, 495, 588]) and on neurobiological aspects (textbooks: [320, 504]). In order to achieve illumination invariance, the retina and the Lateral Geniculate Nucleus (LGN) are essential: these brain areas adjust the high dynamic range of the incoming signals to the limited activity range of subsequent neurons. Luminance and contrast information are processed by separate mechanisms referred to as luminance control and contrast gain control, respectively [81]. As models of these mechanisms, divisive normalization techniques (also referred to as divisive inhibition) have been established in the literature (examples: [50, 51, 162, 192, 276, 406, 487]; review: [81]). The models include a division of the raw or preprocessed input signal by the *local luminance*

$$\mathrm{L}(\boldsymbol{I}) = \boldsymbol{K}_{\mathtt{gau}}(\sigma) * \boldsymbol{I}, \tag{5.13}$$

by the *local intensity variation*

$$\mathrm{V}(\boldsymbol{I}) = \sqrt{\boldsymbol{K}_{\mathtt{gau}}(\sigma) * (\boldsymbol{I} - \mathrm{L}(\boldsymbol{I}))^2}, \tag{5.14}$$

or by a combination of both. The local luminance $\mathrm{L}(\boldsymbol{I})$ is the locally weighted average of gray-values; the local intensity variation $\mathrm{V}(\boldsymbol{I})$ is the locally weighted deviation from the local luminance $\mathrm{L}(\boldsymbol{I})$. Both building blocks for divisive normalization methods operate locally on image patches rather than on the entire image. The size of the considered patches depends on the standard deviation σ of the involved Gaussian filter kernel $K_{\mathtt{gau}}(\sigma)$. This also reduces the influence of signal noise which would otherwise be amplified. Divisive normalization techniques are closely related to image whitening, i.e. removing correlations between neighboring pixels (e.g. [233, 494, 495]).

Based on a divisive normalization, we test the following early-vision models: the *local-contrast function*

$$\mathrm{p}_{\mathtt{dl}}(\boldsymbol{I}) = \frac{\boldsymbol{V}(\boldsymbol{I})}{\boldsymbol{L}(\boldsymbol{I})}, \tag{5.15}$$

was proposed by MANTE et al. [406] and computes the local contrast of the considered image patch. In this case, the divisive normalization is obtained through dividing the local intensity variation $\mathrm{V}(\boldsymbol{I})$ by the local luminance $\mathrm{L}(\boldsymbol{I})$. The function

$$\mathrm{p}_{\mathtt{dv}}(\boldsymbol{I}) = \frac{\boldsymbol{I} - \boldsymbol{L}(\boldsymbol{I})}{1 - \frac{\boldsymbol{V}(\boldsymbol{I})}{\sigma_{1/2}}}, \tag{5.16}$$

referred to as *dividing by variation* was proposed by Stürzl and Zeil [612]. It divides the deviation of the intensity $\boldsymbol{I}$ from the local luminance $\mathrm{L}(\boldsymbol{I})$ by the local intensity variation $\mathrm{V}(\boldsymbol{I})$. The strength of the divisive normalization is adjusted by the additional parameter $\sigma_{1/2}$ with larger values of $\sigma_{1/2}$ resulting in a stronger normalization. As the difference in the function's numerator can get negative, this function can obtain negative pixel intensities, and special care was required for implementation (section 5.3.1.3). The *dividing by luminance/contrast* function

$$\mathrm{p}_{\mathtt{dlc}}(\boldsymbol{I}) = \frac{\boldsymbol{I}}{\boldsymbol{L}(\boldsymbol{I})\left(1 + \frac{\boldsymbol{V}(\boldsymbol{I})}{\boldsymbol{L}(\boldsymbol{I})}\right)} \tag{5.17}$$

was derived by Böttcher [60]. It normalizes the image intensities $\boldsymbol{I}$ depending on the local luminance $\mathrm{L}(\boldsymbol{I})$ and the local contrast $\frac{\mathrm{V}(\boldsymbol{I})}{\mathrm{L}(\boldsymbol{I})}$. Therefore, it strengthens dark image regions and image areas with low local contrast similar to properties of the early visual system discussed in (e.g. [162]).

Histogram-Based Preprocessing

The preprocessing function $\mathrm{p}_{\mathtt{heq}}$ performs a *histogram equalization* (e.g. [67, 241]) of the image $\boldsymbol{I}$. Hence, it adjusts the global image contrast by transforming the image's gray-value distribution into a uniform distribution. The uniform distribution is computed w.r.t. a certain number b of histogram bins.

5.2.2. Global Image Dissimilarity Functions

Within the compass function $\mathrm{z_d}$ (equation (5.2)), the image dissimilarity function d is used to compare the preprocessed images $\boldsymbol{P} = \mathrm{p}(\boldsymbol{I}_P)$ and $\boldsymbol{Q} = \mathrm{p}(\boldsymbol{I}_Q)$. Please refer to [19, 91, 92, 225, 549, 635] for reviews on different image dissimilarity functions. For image comparisons, image dissimilarity functions rely on pixel-by-pixel operations and should —in an ideal case— exhibit the following four properties:

$$\mathrm{d}(\boldsymbol{P},\boldsymbol{Q}) \geq 0 \tag{5.18}$$

$$\mathrm{d}(\boldsymbol{P},\boldsymbol{Q}) = 0 \text{ iff } \boldsymbol{P} = \boldsymbol{Q} \tag{5.19}$$

$$\mathrm{d}(\boldsymbol{P},\boldsymbol{Q}) = \mathrm{d}(\boldsymbol{Q},\boldsymbol{P}) \tag{5.20}$$

$$\mathrm{d}(\boldsymbol{P},\boldsymbol{Q}) \mathrel{\dot{\sim}} \mathrm{dist}(\boldsymbol{x}_P,\boldsymbol{x}_Q) \tag{5.21}$$

Equations (5.18) and (5.19) require the dissimilarity function d to be positive and zero if and only if the images $\boldsymbol{P}$ and $\boldsymbol{Q}$ are identical. In their usual definition, correlation-based dissimilarity functions (equations (5.36) to (5.39)) and the mutual information (equation (5.42)) yield a maximum value for identical images. Nevertheless, by considering the negative of these, this maximization can be turned into a minimization (the definitions given below follow this principle). This allows to apply the compass function (equation (5.2)) without further adaptation. However, equations (5.18) and (5.19) have to be modified to take into account that for these comparison functions (i) a possibly negative global minimum not equal to zero should be taken for identical images and (ii) a comparison of different images should lead to values larger than this global minimum. Equation (5.20) requires the dissimilarity function d to be symmetric; equation (5.21) states that the image dissimilarity $\mathrm{d}(\boldsymbol{P},\boldsymbol{Q})$ should depend monotonically on the spatial distance $\mathrm{dist}(\boldsymbol{x}_p,\boldsymbol{x}_q)$ between the positions of image acquisition $\boldsymbol{x}_p$ and $\boldsymbol{y}_p$.

Within this chapter, we only consider *global* image dissimilarity functions which compare the entire image pixel-by-pixel without sub-dividing it into several sub-images as it is the case for local dissimilarity functions. *Local* dissimilarity functions compute the dissimilarity for each of the subpatches and combine the obtained sub-dissimilarities to a scalar dissimilarity value. Here, we do not consider local image dissimilarity functions because dividing the image into patches and fusing

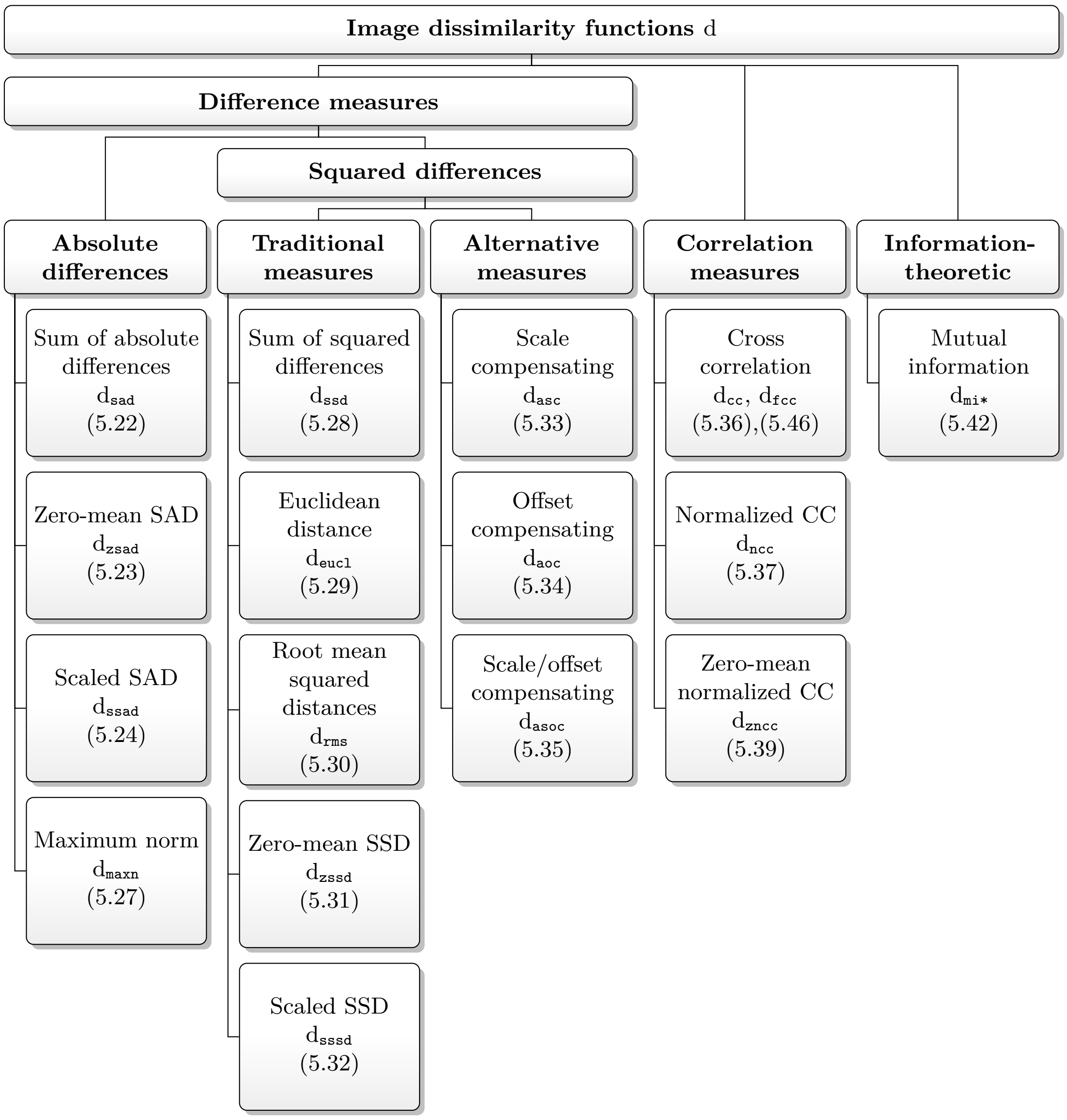


Image dissimilarity functions d
Difference measures
Squared differences
Absolute differences
Traditional measures
Alternative measures
Correlation measures
Information-theoretic
Sum of absolute differences d_{sad} (5.22)
Zero-mean SAD d_{zsad} (5.23)
Scaled SAD d_{ssad} (5.24)
Maximum norm d_{maxn} (5.27)
Sum of squared differences d_{ssd} (5.28)
Euclidean distance d_{eucl} (5.29)
Root mean squared distances d_{rms} (5.30)
Zero-mean SSD d_{zssd} (5.31)
Scaled SSD d_{sssd} (5.32)
Scale compensating d_{asc} (5.33)
Offset compensating d_{aoc} (5.34)
Scale/offset compensating d_{asoc} (5.35)
Cross correlation d_{cc}, d_{fcc} (5.36),(5.46)
Normalized CC d_{ncc} (5.37)
Zero-mean normalized CC d_{zncc} (5.39)
Mutual information d_{mi*} (5.42)

Figure 5.4.: Categorization of tested image dissimilarity functions. Dissimilarity functions tested with several parameters are marked by a star (*); the numbers given in parentheses refer to the equation defining the corresponding function. Extended after Giachetti [225].

the dissimilarities of the patches would increase the computational complexity of the comparison methods. Furthermore, the probability of ambiguous matches is larger for comparing smaller image regions.

In order to achieve robust combinations of image preprocessing and image dissimilarity functions, we test various dissimilarity functions (figure 5.4) based on absolute gray-value differences (d_{sad}, d_{zsad}, d_{ssad} and d_{maxn}), based on squared gray-value differences (d_{ssd}, d_{eucl}, d_{rms}, d_{zssd}, d_{sssd}, d_{asc}, d_{aoc} and d_{asoc}), based on image correlations (d_{cc}, d_{fcc}, d_{ncc}, and d_{zncc}) and based on an information-theoretic measure (d_{mi}). These measures will be described in the following.

Dissimilarity Functions Using Absolute Pixel Differences

Difference measures use the difference between corresponding pixels to measure the image dissimilarity. In order to achieve a symmetric measure and to achieve only positive distances, the measures rely on sums of absolute or squared pixel differences. For the class of difference measures relying on absolute distances, the *sum of absolute differences*

$$\mathrm{d}_{\mathtt{sad}}(\boldsymbol{P},\boldsymbol{Q}) = \sum_{x=0}^{w-1}\sum_{y=0}^{h-1} |P(x,y) - Q(x,y)| \tag{5.22}$$

adds up the absolute values of all pixel differences. The functions

$$\mathrm{d}_{\mathtt{zsad}}(\boldsymbol{P},\boldsymbol{Q}) = \mathrm{d}_{\mathtt{sad}}(\boldsymbol{P} - \bar{P}, \boldsymbol{Q} - \bar{Q}) \text{ and} \tag{5.23}$$

$$\mathrm{d}_{\mathtt{ssad}}(\boldsymbol{P},\boldsymbol{Q}) = \mathrm{d}_{\mathtt{sad}}\left(\boldsymbol{P}, \frac{\bar{P}}{\bar{Q}}\boldsymbol{Q}\right) \tag{5.24}$$

are variants of $\mathrm{d}_{\mathtt{sad}}$: the *zero-mean sum of absolute differences* $\mathrm{d}_{\mathtt{zsad}}$ subtracts the average pixel intensities

$$\bar{P} = \frac{1}{wh}\sum_{x=0}^{w-1}\sum_{y=0}^{h-1} P(x,y) \text{ and} \tag{5.25}$$

$$\bar{Q} = \frac{1}{wh}\sum_{x=0}^{w-1}\sum_{y=0}^{h-1} Q(x,y), \tag{5.26}$$

and the *scaled sum of squared differences* $\mathrm{d}_{\mathtt{ssad}}$ scales the image with the ratio $\frac{\bar{P}}{\bar{Q}}$ of average pixel intensities. The latter function violates equation (5.20) because scaling the intensities of only one image results in a non-symmetric dissimilarity measure. It could be turned into a symmetric measure by scaling both images as proposed by MÖLLER [449], but this approach is not considered here. As the average pixel intensities are independent of the current shift s, $\bar{P}$ and $\bar{Q}$ do not have to be recomputed for each shift s considered by the compass method.

The *maximum norm*

$$\mathrm{d}_{\mathtt{maxn}}(\boldsymbol{P},\boldsymbol{Q}) = \max_{x,y} |P(x,y) - Q(x,y)| \tag{5.27}$$

computes the maximum of all absolute pixel differences.

Dissimilarity Functions Using Squared Pixel Differences

Functions relying on squared pixel differences are all variants of the *sum of squared differences function*

$$\mathrm{d}_{\mathtt{ssd}}(\boldsymbol{P},\boldsymbol{Q}) = \sum_{x=0}^{w-1}\sum_{y=0}^{h-1} \left(P(x,y) - Q(x,y)\right)^2. \tag{5.28}$$

The variants can further be categorized into traditional dissimilarity measures and alternative dissimilarity measures. Traditional variants are the *Euclidean distance* $\mathrm{d}_{\mathtt{eucl}}$ obtained by taking the square root, the *root mean square error* $\mathrm{d}_{\mathtt{rms}}$ obtained taking the square root of the average squared pixel differences, the *zero-mean sum of squared differences* $\mathrm{d}_{\mathtt{zssd}}$ obtained by subtracting the average pixel intensities $\bar{P}$ and $\bar{Q}$ from the images, and the *scaled sum of squared differences* $\mathrm{d}_{\mathtt{sssd}}$ obtained by scaling the image $\boldsymbol{Q}$ by $\frac{\bar{P}}{\bar{Q}}$:

$$\mathrm{d}_{\mathtt{eucl}}(\boldsymbol{P},\boldsymbol{Q}) = \sqrt{\mathrm{d}_{\mathtt{ssd}}(\boldsymbol{P},\boldsymbol{Q})}, \tag{5.29}$$

$$\mathrm{d}_{\mathtt{rms}}(\boldsymbol{P},\boldsymbol{Q}) = \sqrt{\frac{1}{wh}\mathrm{d}_{\mathtt{ssd}}(\boldsymbol{P},\boldsymbol{Q})}, \tag{5.30}$$

$$\mathrm{d}_{\mathtt{zssd}}(\boldsymbol{P},\boldsymbol{Q}) = \mathrm{d}_{\mathtt{ssd}}\left(\boldsymbol{P} - \bar{P}, \boldsymbol{Q} - \bar{Q}\right) \text{ and} \tag{5.31}$$

$$\mathrm{d}_{\mathtt{sssd}}(\boldsymbol{P},\boldsymbol{Q}) = \mathrm{d}_{\mathtt{ssd}}\left(\boldsymbol{P}, \frac{\bar{P}}{\bar{Q}}\boldsymbol{Q}\right). \tag{5.32}$$

The Euclidean distance $\mathrm{d}_{\texttt{eucl}}$ and the root mean square error $\mathrm{d}_{\texttt{rms}}$ are tested only for the sake of the completeness. They are scaled versions of the sum of squared differences and therefore take identical optima. The scaled sum of squared differences $\mathrm{d}_{\texttt{sssd}}$ is not a symmetric measure and therefore violates equation (5.20). Symmetry could be achieved by scaling both images as proposed in Möller [449].

Alternative Dissimilarity Functions

In Möller [448, 449] a set of alternative dissimilarity measures[2] are proposed which are based on the sum of squared differences measure but were derived in order to tolerate certain changes of the illumination. The methods were originally developed for 2D-warping methods (sections 3.5.2.2 and 4.4.2) and are here applied as global image dissimilarity functions. To derive the measures, it is assumed that changes of the illumination can be modeled as a linear transformation of image intensities, i.e. as a scaling and a constant shift of the image intensity (section 5.2.4). By minimizing the influence of such an intensity change onto the resulting sum of squared distances measure, a set of dissimilarity functions

$$\mathrm{d}_{\texttt{asc}}(\boldsymbol{P},\boldsymbol{Q}) = \underbrace{\|\boldsymbol{P}\|\|\boldsymbol{Q}\|}_{r.i.} - \underbrace{\langle\boldsymbol{P},\boldsymbol{Q}\rangle}_{r.d.}, \tag{5.33}$$

$$\mathrm{d}_{\texttt{aoc}}(\boldsymbol{P},\boldsymbol{Q}) = wh\left(\underbrace{\frac{\operatorname{var}(\boldsymbol{P}) + \operatorname{var}(\boldsymbol{Q})}{2}}_{r.i.} - \underbrace{\operatorname{cov}(\boldsymbol{P},\boldsymbol{Q})}_{r.d.}\right), \text{ and} \tag{5.34}$$

$$\mathrm{d}_{\texttt{asoc}}(\boldsymbol{P},\boldsymbol{Q}) = wh\left(\underbrace{\sqrt{\operatorname{var}(\boldsymbol{P})\operatorname{var}(\boldsymbol{Q})}}_{r.i.} - \underbrace{\operatorname{cov}(\boldsymbol{P},\boldsymbol{Q})}_{r.d.}\right) \tag{5.35}$$

is obtained (see appendix C.1 for a recapitulation of the derivation). Depending on the type of intensity changes which can be compensated by the dissimilarity function, we refer to these functions as *scale-compensating function* ($\mathrm{d}_{\texttt{asc}}$), *offset compensating function* ($\mathrm{d}_{\texttt{aoc}}$) and *scale/offset-compensating function* ($\mathrm{d}_{\texttt{asoc}}$).

As the functions do not include a divisive normalization (as is the case for the early-vision models), these functions do not amplify noise contained in homogeneous image regions. If applied as dissimilarity function for the visual compass, parts of the functions are rotationally invariant (in equations (5.33) to (5.35) marked by *r.i.*) and can be precomputed before the image is shifted. The rotation-dependent components (marked by *r.d.*) have to be computed for each tested image shift. In equations (5.33) to (5.35), the notation of the scalar product, the norm, the variance, and the covariance are for sake of simplicity extended from vectors to images because every image can easily be converted into a vector, e.g. by stacking image columns. For matching image columns in the 2D-warping algorithm, the dissimilarity functions $\mathrm{d}_{\texttt{asc}}$, $\mathrm{d}_{\texttt{aoc}}$, and $\mathrm{d}_{\texttt{asoc}}$ are combined with a dissimilarity function measuring the squared differences of the mean image intensities (Möller [448, 449]). This measure is not invariant against changes of the illumination and was introduced in order to avoid mismatches between image columns resulting from erroneously transforming the intensities of one image column into those of the other. In the context of 2D warping, such mismatches can for example occur for comparing two image columns of constant, but different intensities. Nevertheless, for pixel-based loop-closure detection we test the proposed dissimilarity measures $\mathrm{d}_{\texttt{asc}}$, $\mathrm{d}_{\texttt{aoc}}$, and $\mathrm{d}_{\texttt{asoc}}$ without such a combination: First, for global image comparisons, the mean image intensities are independent of the robot's orientation and thus the combined distance measure would only have an influence on the minimum image dissimilarity $\hat{\ell}$ but not on the compass shift $\hat{s}$. Second, we consider such erroneous mismatches to be less likely if entire images are compared instead of single image columns.

[2]To make explicit that these measures belong to the same class, their acronyms all begin with an "a" for "alternative".

Correlation-Based Dissimilarity Functions

All correlation-based dissimilarity functions are variants of the negative *cross-correlation function*

$$\mathrm{d}_{\texttt{cc}}(\boldsymbol{P},\boldsymbol{Q}) = -\underbrace{\langle \boldsymbol{P},\boldsymbol{Q}\rangle}_{r.d.}. \tag{5.36}$$

The computational effort of computing the cross-correlation function can be considerably reduced by applying the cross-correlation theorem and computing the cross correlation in the Fourier domain (textbook: [65]). Section 5.2.3 describes a variant of the visual compass method approximating $\mathrm{d}_{\texttt{cc}}$ based on this theorem. The variants of the cross-correlation function include the *normalized cross-correlation coefficient*

$$\mathrm{d}_{\texttt{ncc}}(\boldsymbol{P},\boldsymbol{Q}) = \underbrace{\frac{1}{\|\boldsymbol{P}\|\|\boldsymbol{Q}\|}}_{r.i.}\mathrm{d}_{\texttt{cc}}(\boldsymbol{P},\boldsymbol{Q}) \tag{5.37}$$

$$= \frac{\mathrm{d}_{\texttt{cc}}(\boldsymbol{P},\boldsymbol{Q})}{\sqrt{\sum_{x=0}^{w-1}\sum_{y=0}^{h-1} P(x,y)^2 \sum_{x=0}^{w-1}\sum_{y=0}^{h-1} Q(x,y)^2}} \tag{5.38}$$

and the *zero-mean normalized cross-correlation coefficient* (again $\bar{P}$ and $\bar{Q}$ are independent of the considered image shift s):

$$\mathrm{d}_{\texttt{zncc}}(\boldsymbol{P},\boldsymbol{Q}) = \mathrm{d}_{\texttt{ncc}}\left(\boldsymbol{P}-\bar{P},\boldsymbol{Q}-\bar{Q}\right) \tag{5.39}$$

$$= \frac{\sum_{x=0}^{w-1}\sum_{y=0}^{h-1}(P(x,y)-\bar{P})(Q(x,y)-\bar{Q})}{\sqrt{\sum_{x=0}^{w-1}\sum_{y=0}^{h-1}(P(x,y)-\bar{P})^2(Q(x,y)-\bar{Q})^2}} \tag{5.40}$$

$$= \frac{\overbrace{\mathrm{cov}(\boldsymbol{P},\boldsymbol{Q})}^{r.d.}}{\underbrace{\sqrt{\mathrm{var}(\boldsymbol{P})\,\mathrm{var}(\boldsymbol{Q})}}_{r.i.}}. \tag{5.41}$$

The normalized cross correlation $\mathrm{d}_{\texttt{ncc}}$ and the zero-mean normalized cross-correlation $\mathrm{d}_{\texttt{zncc}}$ are closely related to the alternative distance measures $\mathrm{d}_{\texttt{asc}}$ and $\mathrm{d}_{\texttt{asoc}}$ (equations (5.33) and (5.35)) because they combine the same building blocks. Relations between different correlation measures are discussed by PAN, XIE, and WANG [502]. Again, the covariance cov between the images is dependent on the robot's orientation, whereas the image variance var is rotationally invariant.

Information-Theoretic Dissimilarity Measure

The *mutual information* $\mathrm{d}_{\texttt{mi}}$ is an information-theoretic measure and computes the mutual dependence of the gray-value distributions underlying the images $\boldsymbol{P}$ and $\boldsymbol{Q}$ (review: [358], textbook: [122]). In contrast to the covariance, the mutual information also measures higher-order dependencies being zero if the images are completely independent of each other. Like for correlation-based dissimilarity measures, the negative of the mutual information

$$\mathrm{d}_{\texttt{mi}}(\boldsymbol{P},\boldsymbol{Q}) = -\sum_{i=0}^{b-1}\sum_{j=0}^{b-1}\underbrace{h_{PQ}(i,j)}_{r.d.}\log\frac{\overbrace{h_{PQ}(i,j)}^{r.d.}}{\underbrace{h_P(i)h_Q(j)}_{r.i.}} \tag{5.42}$$

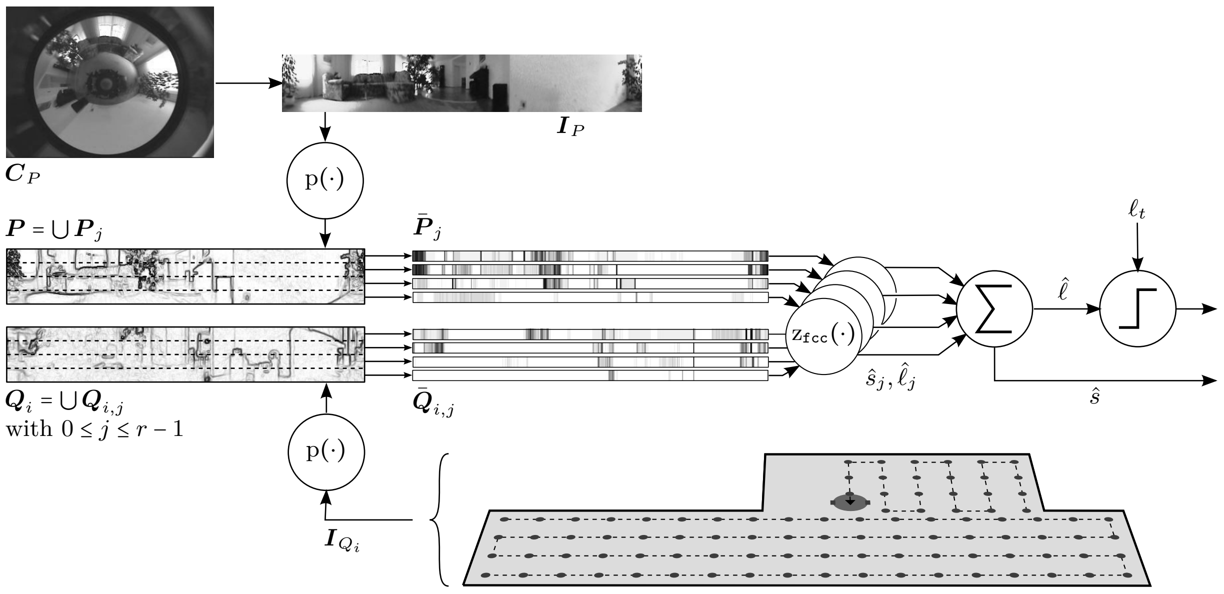


Figure 5.5.: Principles of accelerated compass variant. The robot's current camera image C_P is unfolded resulting in the cylindrical image I_P. I_P is then preprocessed by applying the preprocessing function p and divided into r one-dimensional subpanoramas $\bar{P}_j$. Each subpanorama $\bar{P}_j$ is compared to the corresponding subpanorama $\bar{Q}_{i,j}$ by computing the cross correlation $d_{\mathtt{fcc}}$ in the Fourier domain. Thus, for each of the r subpanoramas, a dissimilarity value $\hat{\ell}_j$ and an estimate of the orientation change $\hat{s}_j$ is computed. The estimates are fused to a scalar dissimilarity value $\hat{\ell}$ used for loop-closure detection and a scalar compass shift $\hat{s}$, respectively.

is used for loop-closure detection because the mutual information reaches its maximum for identical images. The gray-value histograms $\boldsymbol{h}_P$ and $\boldsymbol{h}_Q$ with b bins represent the gray-value distributions of the images $\boldsymbol{P}$ and $\boldsymbol{Q}$; the joint image histogram $\boldsymbol{h}_{PQ}$ approximates the two-dimensional joint distribution representing the probability that corresponding pixels $P(x,y)$ and $Q(x,y)$ belong to bins i and j (for $0 \leq x < w$ and $0 \leq y < h$). As the gray-value histograms $\boldsymbol{h}_P$ and $\boldsymbol{h}_Q$ are independent of the considered image shift s, only the joint histogram $\boldsymbol{h}_{PQ}$ has to be recomputed inside the compass loop. The mutual information has been applied as a distance measure for stereo matching (e.g. [285, 286]), image registration (e.g. [120, 516, 627]), visual homing (e.g. [617]), object tracking (e.g. [503]), and for feature extraction (e.g. [479, 510]).

5.2.3. Accelerated Compass Method Operating in the Fourier Domain

In this section, we describe a new holistic compass method operating in the Fourier domain. The method is capable of estimating the compass shift more efficiently than the standard compass method by Zeil, Hoffmann, and Chahl [718] described at the beginning of figure 5.1. Our new method is more efficient because it computes the compass shift and the image dissimilarity value in the Fourier domain from a single image comparison without shifting one of the images step-by-step. The method is sketched in figure 5.5 and relies on the discrete cross-correlation theorem

$$\mathcal{F}(\boldsymbol{f} \star \boldsymbol{g}) = \mathcal{F}(\boldsymbol{f})^* \cdot \mathcal{F}(\boldsymbol{g}), \tag{5.43}$$

which states that the cross correlation $\boldsymbol{f} \star \boldsymbol{g}$ of two periodic functions $\boldsymbol{f}$ and $\boldsymbol{g}$ can be expressed in the frequency domain as a multiplication of the complex conjugate $\mathcal{F}(\boldsymbol{f})^*$ with $\mathcal{F}(\boldsymbol{g})$ (textbook: [65]). For the special case of the auto-correlation, i.e. for $\boldsymbol{f} = \boldsymbol{g}$, the cross-correlation theorem is referred to as Wiener-Khinchin theorem. The theorem generalizes to two- and more-dimensional functions but also requires the periodicity of the functions in more dimensions. Due to this, the theorem cannot be applied to 2D panoramic images as the images are periodically closed only in horizontal direction, but not in vertical.

To apply the theorem to panoramic images, the preprocessed images $\boldsymbol{P}$ and $\boldsymbol{Q}$ are subdivided into r subpanoramas $\boldsymbol{P}_j$ and $\boldsymbol{Q}_j$ $(0 \leq j < r)$ of equal height. The principle of subpanoramas is also used by Gonzalez-Barbosa and Lacroix [242] for signature-based navigation and is in chapter 6 used for signature-based loop-closure detection. By column-wise averaging, each of these subpanoramas is reduced to a one-dimensional row image referred to as $\bar{\boldsymbol{P}}_j$ for the current image and $\bar{\boldsymbol{Q}}_j$ for the images stored in the topological map, respectively. For each of these one-dimensional images, the cross-correlation theorem is applied to compute the cross-correlation function

$$\boldsymbol{\ell}_{\mathtt{exact},j} = \mathcal{F}^{-1}\left(\mathcal{F}(\bar{\boldsymbol{P}}_j)^* \cdot \mathcal{F}(\bar{\boldsymbol{Q}}_j)\right). \tag{5.44}$$

As most of the relevant image information is contained in the lower-order Fourier coefficients (e.g. [421, 495, 588, 611]), the computation of the cross-correlation function ℓ can be further accelerated by multiplying the first b Fourier coefficients only and assuming higher-frequency components to be zero (with $\mathcal{F}_b$ denoting the discrete Fourier transformation restricted to b coefficients):

$$\begin{aligned}\boldsymbol{\ell}_j &= \mathrm{d}_{\mathtt{fcc}}\left(\bar{\boldsymbol{P}}_j, \bar{\boldsymbol{Q}}\right) &(5.45)\\ &= \mathcal{F}^{-1}\left(\mathcal{F}_b(\bar{\boldsymbol{P}}_j)^* \cdot \mathcal{F}_b(\bar{\boldsymbol{Q}}_j)\right). &(5.46)\end{aligned}$$

In contrast to the standard method computing dissimilarities in the image domain (equation (5.1)), equation (5.46) yields the w-dimensional vector $\boldsymbol{\ell}$ of dissimilarity values in a single step. To obtain a cross-correlation function with w elements in the image domain, all coefficients —and not only the first b coefficients— have to be considered for the inverse Fourier transformation $\mathcal{F}^{-1}$. We refer this function as *approximated cross-correlation function* $\mathrm{d}_{\mathtt{fcc}}$. Thereupon, the change of orientation

$$\hat{s}_j = \arg \max_{s=0}^{w-1} \ell_j(s) \tag{5.47}$$

and the dissimilarity value

$$\begin{aligned}\hat{\ell}_j &= \max_{s=0}^{w-1} \ell_j(s) &(5.48)\\ &= \ell_j(\hat{s}_j) &(5.49)\end{aligned}$$

are computed. In order to fuse the estimates obtained from each of the r subpanoramas,

$$\begin{aligned}\hat{s} &= \operatorname{median}\left(\hat{s}_0, \hat{s}_1, \dots, \hat{s}_{r-1}\right) \text{ and} &(5.50)\\ \hat{\ell} &= \sum_{j=0}^{r-1} \hat{\ell}_j &(5.51)\end{aligned}$$

are computed. Based on $\hat{\ell}$, a binary classification whether or not the images $\boldsymbol{I}_P$ and $\boldsymbol{I}_Q$ were taken at identical positions is performed. The compass shift $\hat{s}$ is an estimate for the robot's change of orientation between capturing images $\boldsymbol{I}_P$ and $\boldsymbol{I}_Q$.

As discussed in section 5.2, the complexity of the standard compass is $\mathcal{O}(w^2 h)$. Computing the cross correlation in the frequency domain results in a complexity of $\mathcal{O}(rw \log w)$ with $\mathcal{O}(w \log w)$ being the complexity of the Fourier-transformation algorithm applied to each of the r subpanoramas. Thus, the accelerated compass variant is considerably faster for small r and large w. This analysis neglects the complexity of computing the row images ($\mathcal{O}(wh)$) before applying the compass method, of computing $\hat{s}_j$ and $\hat{\ell}_j$ ($\mathcal{O}(rw)$), and of fusing them to scalar estimates $\hat{s}$ and $\hat{\ell}$ ($\mathcal{O}(r)$). A more detailed evaluation of the algorithm's computing time is given in section 5.5.3.

The holistic visual compass methods proposed by [75, 610] are closely related to our method. The method by Burke and Vardy [75] also applies the cross-correlation theorem to efficiently compare images. However, it does not limit the number of Fourier coefficients and the authors test their methods either for computing a correlation value for every row of the image or for averaging over image columns and computing a single correlation value for a 1D image. The method by Stürzl and Mallot [610] does not apply the cross-correlation theorem to compute the compass shift but rather uses a coarse-to-fine search for the maximum correlation. This search process —like our method— uses only a reduced number of Fourier coefficients. For further on related work please refer to section 3.4.2.1.

5.2.4. Robustness Against Changes of the Illumination

Changes of the illumination conditions can considerably alter the visual appearance of the perceived omnidirectional images (section 3.2.3.2). Exactly modeling these changes by image-rendering techniques requires exact geometrical and physical models of the robot's workspace and a physical model of the illumination conditions. For an introduction into these techniques, the reader is referred to computer graphics textbooks (e.g. [47, 184]). As a simplification, the influence of illumination changes onto the image intensities can be modeled as a linear transformation

$$\boldsymbol{I}' = a\boldsymbol{I} + o. \tag{5.52}$$

of the pixel intensities with a being the scale factor and o being the intensity offset (e.g. [438, 482]). Analyzing

$$\mathrm{d}(\mathrm{p}(\boldsymbol{I}), \mathrm{p}(a\boldsymbol{I} + o)) \tag{5.53}$$

allows then to draw theoretical conclusions how robust the tested combinations of preprocessing functions p and dissimilarity function d are against changes of the illumination. In an ideal case, the combination of preprocessing function p and dissimilarity function d could completely compensate for the intensity changes resulting from illumination changes. Then, the following relation would hold:

$$\begin{aligned}\mathrm{d}(\mathrm{p}(\boldsymbol{I}), \mathrm{p}(a\boldsymbol{I} + o)) &= \mathrm{d}(\mathrm{p}(\boldsymbol{I}), \mathrm{p}(1\boldsymbol{I} + 0)) &(5.54)\\ &= \mathrm{d}(\mathrm{p}(\boldsymbol{I}), \mathrm{p}(\boldsymbol{I})) &(5.55)\end{aligned}$$

5.2.4.1. Influence of Illumination Changes onto Image Preprocessing

For this purpose, we first analyze how preprocessing functions are influenced by a linear intensity transformation

$$\boldsymbol{P}' = \mathrm{p}(a\boldsymbol{I} + o) \tag{5.56}$$

of the input image. Because most considered preprocessing functions are linear, we can analyze

$$\boldsymbol{P}' = a\,\mathrm{p}(\boldsymbol{I}) + o \tag{5.57}$$

instead of equation (5.56). The results of the first evaluation step are summarized in table 5.1.1, and a mathematical derivation of these results is given in appendix C.2.

None of the preprocessing functions can completely compensate for such a linear intensity transformation. The functions $\mathrm{p}_{\mathtt{pw}}$, $\mathrm{p}_{\mathtt{sob}}$, $\mathrm{p}_{\mathtt{lap}}$, $\mathrm{p}_{\mathtt{pw}}$, $\mathrm{p}_{\mathtt{dog}}$, and $\mathrm{p}_{\mathtt{dv}}$ preserve the scaling a and compensate for the intensity shift o. For the functions $\mathrm{p}_{\mathtt{dl}}$ and $\mathrm{p}_{\mathtt{dlc}}$, preprocessing the transformed image $\boldsymbol{P}' = a\boldsymbol{I} + o$ is identical to preprocessing an image $\boldsymbol{P}' = \boldsymbol{I} + \frac{o}{a}$ undergoing an intensity shift of $\frac{o}{a}$ (equations (C.36) and (C.43)). Hence, in case the image intensities are only scaled (i.e. $o = 0$), these preprocessing functions compensate for the scaling a. As $\mathrm{p}_{\mathtt{id}}$ is the identity function, it preserves both the scaling and the intensity shift (i.e. $\mathrm{p}_{\mathtt{id}}(a\boldsymbol{I} + o) = a\boldsymbol{I} + o$). Due to the nonlinearities and the binning involved in computing the histogram equalization $\mathrm{p}_{\mathtt{heq}}$, we cannot draw reasonable conclusions for this preprocessing function.

Table 5.1.: Robustness against linear changes of the image intensities. If the parameters a or o of the linear transformation $a\boldsymbol{I} + o$ are compensated by a preprocessing function p (subtable (1)) or an image distance function d (subtable (2)), the corresponding cell of the table is marked by ✓; otherwise it is marked by ✗. Subtable (3) summarizes the results for each combination of preprocessing and image distance functions. Each cell of the table contains whether the scale a and offset o (./.) are compensated (✓) or not (✗). Due to the nonlinearities involved in the histogram equalization $\mathrm{p}_{\mathtt{heq}}$ and the mutual information $\mathrm{d}_{\mathtt{mi}}$, we cannot make reasonable predictions without further assumptions of the image's gray-value distribution. To this end, these entries are in all tables marked by a dash (—).

(1) Preprocessing functions

Parameter	Preprocessing function								
	$\mathrm{p}_{\mathtt{id}}$	$\mathrm{p}_{\mathtt{pw}}$	$\mathrm{p}_{\mathtt{sob}}$	$\mathrm{p}_{\mathtt{lap}}$	$\mathrm{p}_{\mathtt{dog}}$	$\mathrm{p}_{\mathtt{heq}}$	$\mathrm{p}_{\mathtt{dl}}$	$\mathrm{p}_{\mathtt{dv}}$	$\mathrm{p}_{\mathtt{dlc}}$
Scale a	✗	✗[1]	✗[1]	✗[1]	✗[1]	—	✓	✗[1]	✓
Offset o	✗	✓	✓	✓	✓	—	✗[2]	✓	✗[2]

[1] The offset o is compensated and $\mathrm{p}(a\boldsymbol{I} + o) = a\,\mathrm{p}(\boldsymbol{I})$.

[2] The scale a is compensated and $\mathrm{p}(a\boldsymbol{I} + o) = \mathrm{p}(\boldsymbol{I} + \frac{a}{o})$.

(2) Image distance functions

Parameter	Dissimilarity function										
	$\mathrm{d}_{\mathtt{sad}}$	$\mathrm{d}_{\mathtt{zsad}}$	$\mathrm{d}_{\mathtt{ssad}}$	$\mathrm{d}_{\mathtt{ssd}}$	$\mathrm{d}_{\mathtt{zssd}}$	$\mathrm{d}_{\mathtt{sssd}}$	$\mathrm{d}_{\mathtt{asc}}$	$\mathrm{d}_{\mathtt{aoc}}$	$\mathrm{d}_{\mathtt{asoc}}$	$\mathrm{d}_{\mathtt{cc}}$	$\mathrm{d}_{\mathtt{ncc}}$	$\mathrm{d}_{\mathtt{mi}}$
	$\mathrm{d}_{\mathtt{maxn}}$			$\mathrm{d}_{\mathtt{eucl}}$					$\mathrm{d}_{\mathtt{zncc}}$	$\mathrm{d}_{\mathtt{fcc}}$		
				$\mathrm{d}_{\mathtt{rms}}$								
Scale a	✗	✗	✗[1]	✗	✗	✗[1]	✓	✗	✓	✗	✗[1]	—
Offset o	✗	✓	✗	✗	✓	✗	✗	✓	✓	✗	✗	—

[1] In case the image intensities are only scaled (i.e. $o = 0$), the scale a is compensated: $\mathrm{d}(\boldsymbol{P}, a\boldsymbol{P}) = \mathrm{d}(\boldsymbol{P}, \boldsymbol{P})$.

(3) Combination of both

Preproc.	Dissimilarity function										
	$\mathrm{d}_{\mathtt{sad}}$	$\mathrm{d}_{\mathtt{zsad}}$	$\mathrm{d}_{\mathtt{ssad}}$	$\mathrm{d}_{\mathtt{ssd}}$	$\mathrm{d}_{\mathtt{zssd}}$	$\mathrm{d}_{\mathtt{sssd}}$	$\mathrm{d}_{\mathtt{asc}}$	$\mathrm{d}_{\mathtt{aoc}}$	$\mathrm{d}_{\mathtt{asoc}}$	$\mathrm{d}_{\mathtt{cc}}$	$\mathrm{d}_{\mathtt{ncc}}$	$\mathrm{d}_{\mathtt{mi}}$
	$\mathrm{d}_{\mathtt{maxn}}$			$\mathrm{d}_{\mathtt{eucl}}$					$\mathrm{d}_{\mathtt{zncc}}$	$\mathrm{d}_{\mathtt{fcc}}$		
				$\mathrm{d}_{\mathtt{rms}}$								
$\mathrm{p}_{\mathtt{id}}$	✗/✗	✗/✓	✗/✗[1]	✗/✗	✗/✓	✗/✗[1]	✓/✗	✗/✓	✓/✓	✗/✗	✗/✗[1]	—
$\mathrm{p}_{\mathtt{pw}}$, $\mathrm{p}_{\mathtt{sob}}$	✗/✓	✗/✓	✓/✓	✗/✓	✗/✓	✓/✓	✓/✓	✗/✓	✓/✓	✗/✓	✓/✓	—
$\mathrm{p}_{\mathtt{lap}}$, $\mathrm{p}_{\mathtt{dog}}$	✗/✓	✗/✓	✓/✓	✗/✓	✗/✓	✓/✓	✓/✓	✗/✓	✓/✓	✗/✓	✓/✓	—
$\mathrm{p}_{\mathtt{heq}}$	—	—	—	—	—	—	—	—	—	—	—	—
$\mathrm{p}_{\mathtt{dl}}$, $\mathrm{p}_{\mathtt{dlc}}$	✓/✗	✓/✓	✓/✗	✓/✗	✓/✓	✓/✗	✓/✗	✓/✓	✓/✓	✓/✗	✓/✗	—
$\mathrm{p}_{\mathtt{dv}}$	✗/✓	✗/✓	✓/✓	✗/✓	✗/✓	✓/✓	✓/✓	✗/✓	✓/✓	✗/✓	✓/✓	—

[1] In case the image intensities are only scaled (i.e. $o = 0$), the scale a is compensated: $\mathrm{d}(\mathrm{p}(\boldsymbol{I}), \mathrm{p}(a\boldsymbol{I})) = \mathrm{d}(\mathrm{p}(\boldsymbol{I}), \mathrm{p}(\boldsymbol{I}))$.

5.2.4.2. Influence of Illumination Changes onto Image Dissimilarity Functions

In a second step, we analyze equation (5.53), i.e. the influence of applying a dissimilarity function d to a preprocessed image $\boldsymbol{P} = \mathrm{p}(\boldsymbol{I})$ and a variant $\boldsymbol{P}' = \mathrm{p}(a\boldsymbol{I} + o)$ of the same image $\boldsymbol{I}$ which underwent a linear intensity change prior to preprocessing.

The results of this evaluation are summarized in table 5.1.2; a derivation of the results is given in appendix C.3. The zero-mean normalized cross correlation $\mathrm{d}_{\mathtt{zncc}}$ and the alternative distance measure $\mathrm{d}_{\mathtt{asoc}}$ compensating for scale and offset are the only dissimilarity functions which can compensate for both a and o. The brightness shift o is compensated by the functions $\mathrm{d}_{\mathtt{zsad}}$, $\mathrm{d}_{\mathtt{zssd}}$, and $\mathrm{d}_{\mathtt{aoc}}$ whereas the scaling a is compensated only by the $\mathrm{d}_{\mathtt{asc}}$ dissimilarity function. For the special case that the transformed image intensities are only scaled (i.e. $o = 0$), the functions $\mathrm{d}_{\mathtt{ssad}}$, $\mathrm{d}_{\mathtt{sssd}}$ and $\mathrm{d}_{\mathtt{ncc}}$ can compensate for the resulting intensity changes. As the dissimilarity functions

$d_{\mathtt{sad}}$, $d_{\mathtt{maxn}}$, $d_{\mathtt{ssd}}$, $d_{\mathtt{eucl}}$, $d_{\mathtt{rms}}$, $d_{\mathtt{cc}}$ and $d_{\mathtt{fcc}}$ cannot compensate for the parameters a or o, we expect that these functions are most sensitive against changes of the illumination. Due to the logarithms and the binning operations involved in computing the mutual information $p_{\mathtt{mi}}$, we cannot draw conclusions how linear intensity changes affect this preprocessing function.

5.2.4.3. Combined Analysis

As most of the preprocessing functions p and the dissimilarity functions d can only partially compensate for linear intensity transformations, we combine the results obtained in the previous steps (sections 5.2.4.1 and 5.2.4.2) and analyze the influence of linear intensity changes on the tested combinations of preprocessing and dissimilarity function. The combined results are shown in table 5.1.3. Except for the dissimilarity measures $d_{\mathtt{asoc}}$ and $d_{\mathtt{zncc}}$, the influence of the linear intensity transformation can only be compensated by a combination of a preprocessing function p and a dissimilarity function d. The following combinations allow for a complete compensation of scaling a and offset o (in table 5.1.3 these combinations are marked by ✓/✓):

- By combining a preprocessing function compensating for the intensity shift o with a dissimilarity function compensating for the scale a independent of the shift o. In this case, the preprocessing methods $p_{\mathtt{pw}}$, $p_{\mathtt{sob}}$, $p_{\mathtt{lap}}$, $p_{\mathtt{dog}}$, or $p_{\mathtt{dv}}$ have to be combined with the alternative distance measure compensating for the scale $d_{\mathtt{asc}}$.

- By combining a preprocessing function compensating for the intensity shift o with a dissimilarity function compensating for the scale a only for the special case $o = 0$. This is the case for combining one of the preprocessing functions $p_{\mathtt{pw}}$, $p_{\mathtt{sob}}$, $p_{\mathtt{lap}}$, $p_{\mathtt{dog}}$, or $p_{\mathtt{dv}}$ with one of the dissimilarity functions $p_{\mathtt{ssad}}$, $p_{\mathtt{sssd}}$ or $p_{\mathtt{ncc}}$.

- By combining a preprocessing function compensating for the scale a with a dissimilarity function compensating for the offset o. This case is achieved for the preprocessing functions $p_{\mathtt{dl}}$ or $p_{\mathtt{dlc}}$ in conjunction with the dissimilarity functions $d_{\mathtt{zsad}}$, $d_{\mathtt{zssd}}$, or $d_{\mathtt{aoc}}$.

- For the special case that image intensities are only scaled (i.e. $o = 0$), the scaling is compensated for the preprocessing function $p_{\mathtt{id}}$ in conjunction with the dissimilarity functions $d_{\mathtt{ssad}}$, $d_{\mathtt{sssd}}$, and $d_{\mathtt{ncc}}$. Furthermore, all combinations of preprocessing functions and dissimilarity functions which compensate for intensity scalings a but preserve intensity offsets also compensate for a in this case.

- For the special case that image intensities only undergo an intensity offset without a scaling (i.e. $o \neq 0$ and $a = 1$), all combinations compensating o but preserving a compensate for both parameters.

Besides these cases, a number of combinations only compensates for one of the parameters of the linear intensity transformation:

- The scaling a is compensated while the offset o is preserved for combining the local contrast $p_{\mathtt{dl}}$ or the luminance/contrast-dividing function $p_{\mathtt{dlc}}$ with one of the dissimilarity functions $d_{\mathtt{sad}}$, $d_{\mathtt{maxn}}$, $d_{\mathtt{ssad}}$, $d_{\mathtt{ssd}}$, $d_{\mathtt{eucl}}$, $d_{\mathtt{rms}}$, $d_{\mathtt{sssd}}$, $d_{\mathtt{asc}}$, $d_{\mathtt{cc}}$, $d_{\mathtt{fcc}}$, and $d_{\mathtt{ncc}}$. These combinations are marked by ✓/✗ in table 5.1.3.

- The scaling a is preserved and the intensity offset is compensated for the preprocessing functions $p_{\mathtt{pw}}$, $p_{\mathtt{sob}}$, $p_{\mathtt{lap}}$, $p_{\mathtt{dog}}$, and $p_{\mathtt{dv}}$ in combination with the $d_{\mathtt{sad}}$, $d_{\mathtt{maxn}}$, $d_{\mathtt{zsad}}$, $d_{\mathtt{ssd}}$, $d_{\mathtt{eucl}}$, $d_{\mathtt{rms}}$, $d_{\mathtt{zssd}}$, $d_{\mathtt{aoc}}$, or $d_{\mathtt{aoc}}$. In table 5.1.3, these combinations are marked by ✗/✓.

The combinations preserving both scale changes a and intensity offsets o are in table 5.1.3 marked by ✗/✗ and include the distance functions $\mathrm{d}_{\mathtt{sad}}$, $\mathrm{d}_{\mathtt{maxn}}$, $\mathrm{d}_{\mathtt{ssd}}$ $\mathrm{d}_{\mathtt{eucl}}$, $\mathrm{d}_{\mathtt{rms}}$, and $\mathrm{d}_{\mathtt{cc}}$ as well as the accelerated variant of the cross correlation $\mathrm{d}_{\mathtt{fcc}}$ for comparing images without prior preprocessing ($\mathrm{p}_{\mathtt{id}}$). These findings will be compared to the results of the database experiments performed and evaluated as described in the next section.

5.3. Experiments and Evaluation

The main objective of the experiments is to find a promising combination of preprocessing function p and dissimilarity function d which allows for robust loop-closure detection under a wide range of illumination conditions and for different workspaces of the robot. Beside that, the experiments are assigned to reveal the following aspects:

- How do changes of the illumination influence the proposed methods?
- How does the accelerated compass variant perform in comparison to the standard method w.r.t. loop-closure detection and compass accuracy?
- How do the image preprocessing methods influence the loop-closure detection performance and the compass accuracy in comparison to using unprocessed images?
- Can reliable loop-closure detection be achieved for a wide range of workspaces, or does the best combination of p and d depend on the environment the robot is operating in?
- Are the proposed methods efficient enough for loop-closure detection in a real-robot implementation?

The experiments were conducted following the experimental procedure as described in section 5.3.1; the obtained results were evaluated with the performance measures proposed in section 5.3.2.

5.3.1. Experimental Procedure

To assess the performance of the proposed methods, we performed a large number of database experiments (section 5.3.1.1) while systematically varying the parameters of the proposed methods (section 5.3.1.2). Section 5.3.1.3 briefly describes implementation details.

5.3.1.1. Image Databases[3]

In order to run a large number of image comparisons and in order to identify promising approaches, we performed database experiments. For our experiments, we used image databases containing images recorded at a regular grid with a grid distance of $g = 10\,\mathrm{cm}$. These databases resemble the *dense* topo-metric maps built while systematically covering the robot's workspace (section 4.2). Since image databases publicly available on the internet (table 3.3) are only applicable for tasks involving *sparse* topological or topo-metric maps, we did not include these databases in our experiments. The used databases (table 5.2) were collected in several apartments under real illumination conditions. Depending on their size, rooms are covered by several database segments (`moellerX` and `livingX`) or a single database segment (all other databases). Figures B.1 and B.3 show example images taken at the corners of the database segments, and figure B.4 visualizes the locations within the rooms where databased have been collected. Since the collection of the larger image databases took several hours, these databases contain considerable variations of the illumination caused by changes of the daylight or the weather conditions. For the `living` databases, images were collected

[3]This section is an extension of section IV in our conference publication Gerstmayr-Hillen et al. [223].

Table 5.2.: Image databases used for the loop-closure detection experiments. The number of snapshots in x and y direction are given by n_x and n_y, respectively. The databases of the `day|day` group (upper part of the table) were collected during the day under natural illumination conditions; the databases of the `day|night` group are cross-databases (hence their labels start with "`c`") collected at identical spatial positions during the day under natural illumination conditions and during the night under constant artificial illumination. For the `clivingXday` databases, the undisturbed snapshot $\boldsymbol{I}_i$ was taken from the `day|day` database, whereas the disturbed snapshot $\boldsymbol{I}'_j$ was taken from the corresponding `night` database (and vice versa for the `clivingXnight` databases).

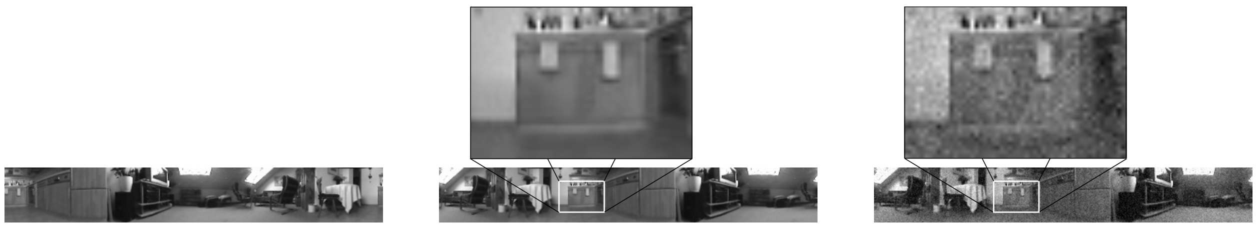
	Database	Abbr.	n_x	n_y	Description	Changes of the illumination
`day|day` group	`kitchen`	`K`	11	8	Tiny kitchen, furniture close-by	None
	`moeller1`	`M1`	22	11	Large living room, white walls	Moderate
	`moeller2`	`M2`	27	19		Considerable
	`roeben`	`R`	36	11	Kitchen/living room	Small
	`living1day`	`L1`	19	5	Mid-sized living room	Small
	`living2day`	`L2`	22	5		
	`living3day`	`L3`	16	11		
	`living4day`	`L4`	15	3		
`day|night` group	`cliving1day`	`C1D`	19	5	Mid-sized living room	Very strong (cross-database test)
	`cliving2day`	`C2D`	22	5		
	`cliving3day`	`C3D`	16	11		
	`cliving4day`	`C4D`	15	3		
	`cliving1night`	`C1N`	19	5	Mid-sized living room	Very strong (cross-database test)
	`cliving2night`	`C2N`	22	5		
	`cliving3night`	`C3N`	16	11		
	`cliving4night`	`C4N`	15	3		

at identical positions under natural illumination conditions during the day and under artificial illumination conditions during the night. By pairing an image from the `livingXday` database with images from the corresponding `livingXnight` database (or vice versa), cross-database experiments simulating drastic changes of the illumination were performed. These databases are named after the database from which the undisturbed snapshot image was taken, and their labels start with a "`c`" to indicate cross-database experiments. For our test application, we consider such drastic changes to be unlikely because we expect the robot to clean the accessible workspace within a maximum of 1 h. Although the illumination conditions of *both* databases building a cross database differ, *each of the two* databases were collected under nearly constant illumination conditions. With the available databases we can simulate that the robot operates under constant illumination conditions and approaches an already cleaned segment which was mapped under very different, but also constant illumination. Thus, the cross-database experiments do not reflect the full variety of different illumination conditions which can occur when illumination continuously changes. Nevertheless, we think that testing the methods with the existing databases is appropriate for assessing the method's robustness against changes of the illumination —especially because many of the proposed methods even fail for the available databases (sections 5.4 and 5.5). Collecting new databases under more realistic illumination conditions should be addressed in future work; for this purpose our new database-collecting tool can be applied (section 3.7.1.1).

For each database, we compared each image $\boldsymbol{I}_i$ against each image $\boldsymbol{I}'_j$ taken from the same database or —in case of cross-database experiments— taken from the corresponding cross-database. Thus, a total of $(n_x n_y)^2$ image dissimilarity values

$$\hat{\ell}_{i,j} = \mathrm{d}\big(\mathrm{p}(\boldsymbol{I}_i), \mathrm{p}(\boldsymbol{I}'_j)\big) \tag{5.58}$$

were computed for each database ($0 \leq i, j < n_x n_y$ with n_x and n_y being the number of images in x- and y-direction). As we aim to develop methods with an optimal performance over many different



(1) Original image **(2)** Rotated **(3)** Rotated and noise-disturbed

Figure 5.6.: Image disturbances used for loop-closure detection. Subfigures (3) to (3) depict the original (undisturbed) image, its rotated variant, and the rotated and noise-disturbed variant. The shown example image was taken at position (18, 5) of the `roeben` database; the noise is Gaussian noise with standard deviation $\sigma = 0.05$.

workspaces, we pool the obtained dissimilarity values into two *test* groups: the `day|day` group pools the databases collected during the day (i.e. `kitchen`, `livingXday`, `moellerX`, and `roeben`) and contains a total of 540,419 image pairings. In the `day|night` group, the cross-databases (i.e. `livingXday` and `livingXnight`) are subsumed. This group contains results from 108,252 image comparisons.

All cylindrical images have a size of 461×64 pixels with intensities in the range $[0, 1]$ and were unfolded from the original camera images by a hyperbolic mapping. For all experiments, the second image $\boldsymbol{I}'_j$ was randomly rotated and disturbed by zero-mean Gaussian noise with standard deviation $\sigma = 0.05$, followed by a thresholding operation to limit the intensities to $[0, 1]$ again (figure 5.6). In order to guarantee identical experimental conditions for all the experiments conducted for pixel-based loop-closure detection (this chapter) and signature-based loop-closure detection (chapter 6), auxiliary image databases were created containing the unfolded (but not preprocessed) images as well as rotated and disturbed variants. These auxiliary databases were used to perform all the loop-closure detection experiments presented in this and the following chapter and are publicly available on the internet [S1].

5.3.1.2. Parameter Variations

To identify promising combinations of preprocessing functions p and dissimilarity functions d, we tested all possible combinations of preprocessing and dissimilarity functions. For the functions depending on internal parameters, we additionally varied these parameters to achieve the best possible performance:

$$b \in \{4, 8, 16, 32, 64\} \quad \text{for } \mathrm{p}_{\mathtt{heq}} \text{ and } \mathrm{d}_{\mathtt{mi}}, \tag{5.59}$$

$$(\sigma_1, \sigma_2) \in \{(0.01, 0.02), (0.01, 0.04), (0.01, 0.08), (0.025, 0.050)\} \quad \text{for } \mathrm{p}_{\mathtt{dog}}, \tag{5.60}$$

$$w' \in \{5, 7, 11, 15\} \quad \text{for } \mathrm{p}_{\mathtt{dl}} \text{ and } \mathrm{p}_{\mathtt{dlc}}, \text{ and} \tag{5.61}$$

$$(w', \sigma_{1/2}) \in \{5, 7, 11, 15\} \times \{0.0, 0.5, 5.0, 50.0\} \quad \text{for } \mathrm{p}_{\mathtt{dv}}. \tag{5.62}$$

Example images resulting from preprocessing the images according to the given parameters are shown in figure 5.7. In the remainder of this chapter, we refer to one concrete choice of preprocessing function, dissimilarity function, and of the corresponding parameters as combination. By varying the parameters as specified above, 740 different combinations were tested.

The standard deviation of the Gaussian filter kernels involved in the early-vision functions $\mathrm{p}_{\mathtt{dl}}$, $\mathrm{p}_{\mathtt{dlc}}$, and $\mathrm{p}_{\mathtt{dv}}$ was chosen to be $\sigma = \frac{w'}{6}$. By choosing σ depending on the patch size w', we assure that the patch tightly fits the Gaussian function without either cropping relevant entries or padding the relevant entries by values close to zero. We refer to the distance and preprocessing functions tested with different parameters by numbering the parameter sets in the order as given above; for

(1) Original image ($p_{\mathtt{id}}$)

(2) Prewitt filter ($p_{\mathtt{pw}}$)

(3) Sobel filter ($p_{\mathtt{sob}}$)

(4) Laplace filter ($p_{\mathtt{lap}}$)

(5) DoG filter ($p_{\mathtt{dog1}}$)

(6) DoG filter ($p_{\mathtt{dog2}}$)

(7) DoG filter ($p_{\mathtt{dog3}}$)

(8) DoG filter ($p_{\mathtt{dog4}}$)

(9) Local contrast ($p_{\mathtt{dl1}}$)

(10) Local contrast ($p_{\mathtt{dl2}}$)

(11) Local contrast ($p_{\mathtt{dl3}}$)

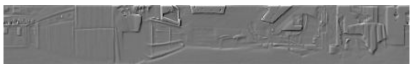
(12) Local contrast ($p_{\mathtt{dl4}}$)

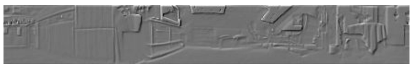
(13) Div. by variation ($p_{\mathtt{dlc1}}$)

(14) Div. by variation ($p_{\mathtt{dlc2}}$)

(15) Div. by variation ($p_{\mathtt{dlc3}}$)

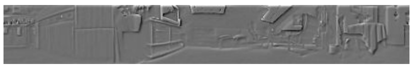
(16) Div. by variation ($p_{\mathtt{dlc4}}$)


(17) Div. by luminance/var. ($p_{\mathtt{dv1/1}}$)


(18) Div. by luminance/var. ($p_{\mathtt{dv1/2}}$)

(19) Div. by luminance/var. ($p_{\mathtt{dv1/3}}$)

(20) Div. by luminance/var. ($p_{\mathtt{dv1/4}}$)


(21) Div. by luminance/var. ($p_{\mathtt{dv2/1}}$)

(22) Div. by luminance/var. ($p_{\mathtt{dv2/2}}$)

(23) Div. by luminance/var. ($p_{\mathtt{dv2/3}}$)

(24) Div. by luminance/var. ($p_{\mathtt{dv2/4}}$)

(25) Div. by luminance/var. ($p_{\mathtt{dv3/1}}$)

(26) Div. by luminance/var. ($p_{\mathtt{dv3/2}}$)

(27) Div. by luminance/var. ($p_{\mathtt{dv3/3}}$)

(28) Div. by luminance/var. ($p_{\mathtt{dv3/4}}$)

(29) Div. by luminance/var. ($p_{\mathtt{dv4/1}}$)

(30) Div. by luminance/var. ($p_{\mathtt{dv4/2}}$)

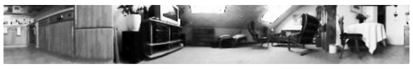
(31) Div. by luminance/var. ($p_{\mathtt{dv4/3}}$)

(32) Div. by luminance/var. ($p_{\mathtt{dv4/4}}$)

(33) Histogram equalization ($p_{\mathtt{heq1}}$)

(34) Histogram equalization ($p_{\mathtt{heq2}}$)

(35) Histogram equalization ($p_{\mathtt{heq3}}$)


(36) Histogram equalization ($p_{\mathtt{heq4}}$)

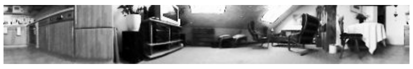

(37) Histogram equalization ($p_{\mathtt{heq5}}$)

Figure 5.7.: Influence of the tested image preprocessing methods. Shown are the original image (1) and preprocessed variants (subimages (2) to (37)) according to the parameters defined in section 5.3.1.2. The image was taken at position $(18, 5)$ of the `roeben` database. For visualization, the images were normalized to $[0, 1]$.

example, we refer to the mutual information with $b = 8$ bins as $\mathrm{d}_{\mathtt{mi2}}$ and to the $\mathrm{p}_{\mathtt{dv}}$ with $w' = 11$ and $\sigma_{1/2} = 50.0$ as $\mathrm{p}_{\mathtt{dv3/4}}$.

The accelerated compass variant was tested in combination with all preprocessing functions p. The number of rings r and the number of Fourier coefficients b were varied in the following ranges:

$$r \in \{1, 2, 4, 8, 16, 32, 64\} \text{ and} \tag{5.63}$$

$$b \in \{4, 8, 16, 32, 64, 96, 128, \text{all}\}, \tag{5.64}$$

where "all" means that the number of Fourier coefficients was not truncated according to equation (5.46). With these parameters, a total of 2072 combinations was tested for the accelerated compass variant. In the remainder of this chapter, we refer to the total number of required Fourier coefficients $r \cdot b$ as "dimensionality". This term is influenced by our work on signature-based loop-closure detection (chapter 6), where it refers to the signature's overall dimensionality. We test such a large number of preprocessing functions p, dissimilarity functions d, and parameters in order to find the best possible combination for our requirements and to allow for a fair comparison. Furthermore, we conduct such an exhaustive comparison because we are not aware of comparable work which we could rely on.

5.3.1.3. Implementation

The methods were implemented in the software framework maintained at the Computer Engineering group by reusing existing building blocks or by extending the framework if required. As the experiments are supposed to be a feasibility study, the implementation was done as a prototype implementation and therefore did not consider optimization possibilities. Computationally more complex processing steps (e.g. image unfolding, preprocessing, visual compass methods including image dissimilarity functions) were implemented in C/C++; the code combining these building blocks in order to generate results was written in Tcl. In this framework, images were represented as arrays of single-precision floating point numbers (e.g. [602]). The software for data evaluation and visualization was implemented by Lorenz Hillen in Matlab (Release 14, Service Pack 2). To compute AUC values, a C program was applied which was developed by Oliver Schlüter during the course of his bachelor's thesis [575] supervised by Lorenz Hillen and Martin Krzykawski. As the output range of the preprocessing function depends strongly on the specific preprocessing functions, the minimum and maximum output was determined by preprocessing all images of the currently used database prior to comparing images. For computing the mutual information, these extreme values were used as lower and upper histogram boundaries for approximating the intensity distribution of the compared images. A small value ϵ close to zero was added to the denominator of preprocessing or dissimilarity functions if required to avoid divisions by zero.

5.3.2. Evaluation

In order to tackle the questions outlined at the beginning of section 5.3, the obtained data was evaluated w.r.t. (i) loop-closure detection performance (section 5.3.2.1), (ii) robustness against perceptual aliasing and perceptual variability (section 5.3.2.2), (iii) accuracy of the involved compass methods (section 5.3.2.3), and (iv) computing time and computational complexity (section 5.3.2.4).

5.3.2.1. Loop-Closure Performance

As a first evaluation step, we assess the loop-closure detection performance by analyzing receiver operator characteristics (ROC). We first introduce this concept and later on describe how it is applied in this thesis.

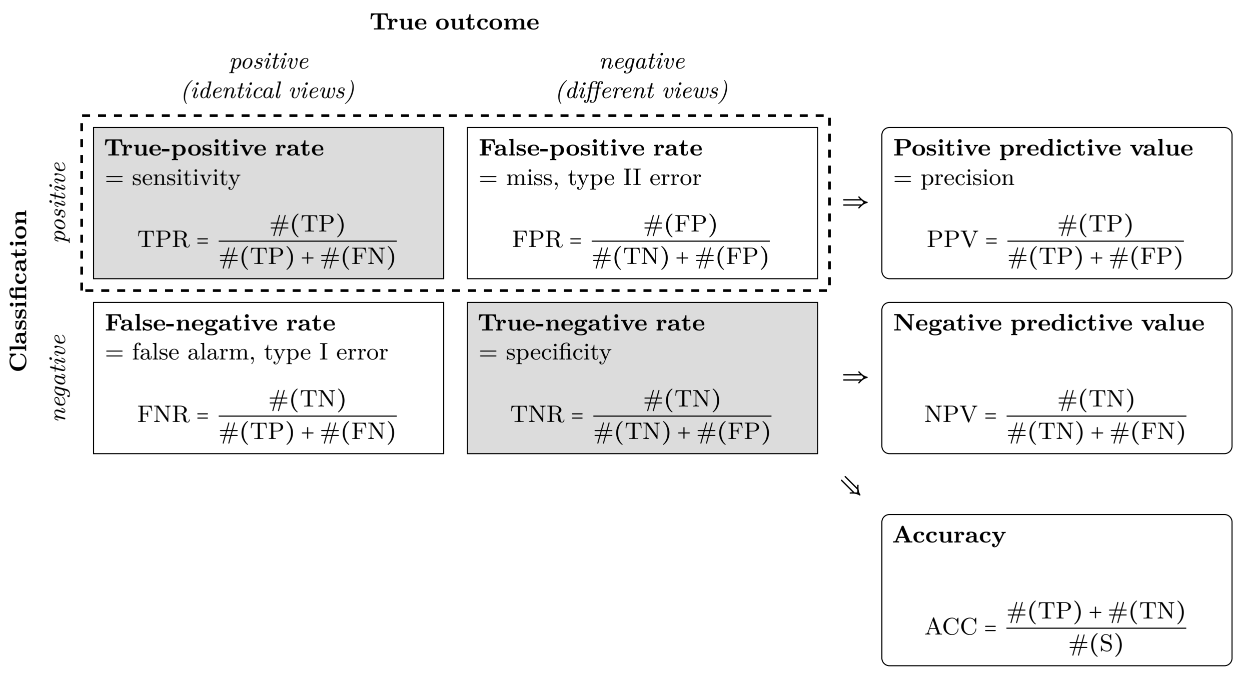

True outcome
positive (identical views)
negative (different views)
Classification
positive
negative
True-positive rate = sensitivity
$\mathrm{TPR} = \frac{\#(\mathrm{TP})}{\#(\mathrm{TP}) + \#(\mathrm{FN})}$
False-positive rate = miss, type II error
$\mathrm{FPR} = \frac{\#(\mathrm{FP})}{\#(\mathrm{TN}) + \#(\mathrm{FP})}$
False-negative rate = false alarm, type I error
$\mathrm{FNR} = \frac{\#(\mathrm{TN})}{\#(\mathrm{TP}) + \#(\mathrm{FN})}$
True-negative rate = specificity
$\mathrm{TNR} = \frac{\#(\mathrm{TN})}{\#(\mathrm{TN}) + \#(\mathrm{FP})}$
Positive predictive value = precision
$\mathrm{PPV} = \frac{\#(\mathrm{TP})}{\#(\mathrm{TP}) + \#(\mathrm{FP})}$
Negative predictive value
$\mathrm{NPV} = \frac{\#(\mathrm{TN})}{\#(\mathrm{TN}) + \#(\mathrm{FN})}$
Accuracy
$\mathrm{ACC} = \frac{\#(\mathrm{TP}) + \#(\mathrm{TN})}{\#(\mathrm{S})}$

Figure 5.8.: Confusion matrix (rectangular boxes) and derived measures of classification performance (boxes with rounded corners). The total number of samples and the number of true-positives, false-positives, true-negatives, and false-negatives samples are denoted by #(S), #(TP), #(FP), #(TN), and #(FN), respectively. Correct classifications (i.e. true-positives and true-negatives) are highlighted by shaded boxes; the true-negative rate and the false-positive rate used for ROC-analysis are grouped by a dashed box. Modified after Fawcett [168].

Introduction to ROC Analysis

Since we used image databases, the true outcome of the classification is known for our experiments, and standard measures for the classification performance as defined in figure 5.8 could be applied. However, these measures do not fulfill our requirements for several reasons:

- All measures are computed for a certain decision threshold ℓ_t. At the current stage of data evaluation, we are rather interested in identifying methods which offer a good general discriminability independent of a certain threshold ℓ_t. We consider choosing a certain threshold to be a further step e.g. required for real-robot experiments.
- According to Provost and Fawcett [523], the measures may be misleading if the class distribution for the classification is skewed. This is the case for our experiments because for each database there are only $n_x n_y$ positives (loop closures) but $(n_x n_y)^2 - n_x n_y$ negatives (images taken at different positions in space).
- In an ideal case, a classifier would yield a true-positive rate of 1 with a false-positive rate of 0. For practical applications, tuning the classifier has to be considered a trade-off between a high true-positive rate and a small false-positive rate as optimizing the true-positive rate usually also increases the false-positive rate (and vice versa) (Provost and Fawcett [523] and Fawcett [168]). Thus, a combination of two measures has to be considered, which complicates the ranking of the classification results obtained for the experiments.

These drawbacks are circumvented by analyzing the classifier's receiver operating characteristic (ROC; [148, 168, 183, 523, 615]) instead of the simple performance measures summarized in figure 5.8. ROC analysis was originally used in the field of signal detection theory (textbooks: [122, 148]) and is now considered to be the standard method for analyzing and comparing the performance of classifiers for a wide range of applications (reviews: [168, 523, 615]).

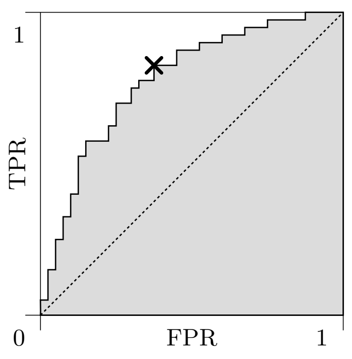


Figure 5.9.: Receiver operating characteristic (ROC). The performance of a classifier with a certain decision threshold ℓ_t can be visualized by plotting its true-positive rate TPR vs. its false-positive rate FPR (cross). By systematically varying the decision threshold ℓ_t, the ROC curve is obtained (solid line). For a random classifier, the ROC curve is identical to the diagonal (dashed). The area under curve (AUC, shaded) is a scalar measure for the classifier's discriminability.

For ROC analysis, the classifier's true-positive rate (TPR) is plotted vs. its false-positive rate (FPR). For a single classifier with a certain decision threshold ℓ_t, this results in a single point in the ROC-space $[0,1] \times [0,1]$ (figure 5.9, cross). By varying the decision threshold ℓ_t, a curve from $(0,0)$ to $(1,1)$ is obtained. For perfect classification, the curve contains the point $(0,1)$; for random classification the curve is identical to the diagonal (figure 5.9, dashed line). Thus, the ROC curve visualizes the classifier's benefits (correct classifications) and the corresponding costs (misclassifications) while implicitly coding ℓ_t along the curve. Due to this, it allows to draw conclusions about the discriminability of the classifier instead of the concrete performance w.r.t. a certain decision threshold [148]. ROC analysis is by [352] applied to evaluate the localization performance of different signature-based approaches. Nevertheless, as the ROC curve is a graphical method, it does not allow to rank or compare a large number of classifiers. For this purpose, the area under the ROC curve is computed by integrating the ROC curve along the x-axis. The resulting scalar value is widely referred to as AUC [168, 523]. The AUC value ranges from 0 (complete misclassification) to 1 (perfect classification) with 0.5 representing chance level. Due to these properties, the AUC value can be used for ranking and comparing the classification results obtained in our experiments. To compute AUC values, we follow the approach by [275]. It sorts the dissimilarity values $\hat{\ell}_{i,j}$ in increasing order and computes the sum S_p of the ranks of positive samples. The AUC can then be computed by

$$\text{AUC} = \frac{S_p - \frac{1}{2}\#(\text{P})(\#(\text{P})+1)}{\#(\text{P})\cdot\#(\text{N})}. \tag{5.65}$$

with $\#(\text{P})$ and $\#(\text{N})$ being the numbers of positive and negative samples, respectively.

Application of ROC Analysis in This Work

The simplest approach for measuring the loop-closure detection performance is to rank all methods of a test group (i.e. of the `day|day` or the `day|night` group) depending on their AUC values. However, this method leads —with the large number of tested parameters— to very long tables and complicates the identification of methods with a good performance for *both* groups. We therefore apply a two-stage evaluation: (i) we identify a small number of methods with a good performance for *both* test groups, and (ii) we analyze AUC values of these methods in detail.

For the first evaluation step, we assign each AUC value to one of the five categories defined in table 5.3. To get an impression of the performance of a group of methods (e.g. for all methods of the `day|day` group comparing images by mutual information and preprocessing images with edge detectors) we compute the frequency of each of the five categories and analyze the resulting histogram. To identify promising methods with a good loop-closure detection for *both* test groups, this approach can be easily extended to the two-dimensional case. By this means, a 5×5 contingency table (or histogram) is obtained which represents the 2D frequency distribution resulting from pairing the categories of the `day|day` group and the `day|night` group. Each cell represents the

Table 5.3.: Categorization of AUC values for analyzing loop-closure detection performance.

Category	Range of AUC values	Description
− − −	$0.00 \le \text{AUC} < 0.60$	Loop-closure detection not possible
∘	$0.60 \le \text{AUC} < 0.90$	Moderate loop-closure detection performance
+	$0.90 \le \text{AUC} < 0.98$	Good performance
++	$0.98 \le \text{AUC} < 1.00$	Very good performance
+ + +	$\text{AUC} = 1.0$	Perfect classification

frequency for a combination of categories, e.g. for perfect classification (+ + +) of the `day|day` group and good classification (+) of the `day|night` group. Instead of identifying combinations with promising performance for both test groups based on the 2D contingency table, it has turned out that it is sufficient to analyze the 1D histograms computed for the `day|day` and the `day|night` group (figures 5.11 and 5.20). The distributions, which are represented by these histograms, correspond to the marginal distributions of the joint 2D distribution, which would only be needed for a more detailed analysis of the possible combinations. For the combinations identified by the proposed approach, we can in the second evaluation step analyze the performance of loop-closure detection in more detail, e.g. by evaluating the AUC values in dependence on the parameters of preprocessing or dissimilarity function. The results of the AUC analysis are presented in sections 5.4.1 and 5.5.1.

5.3.2.2. Robustness Against Perceptual Aliasing and Perceptual Variability

Besides the AUC analysis described in the previous section, we also evaluate the robustness of the proposed methods against perceptual aliasing (also referred to as "spatial aliasing") and perceptual variability. Perceptual aliasing occurs if two images $\boldsymbol{I}_i$ and $\boldsymbol{I}'_j$ taken at different positions $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in space have the same visual appearance (equations (5.5), (5.19) and (5.21)). Perceptual variability is caused by changes of the visual appearance resulting from dynamic scene changes or from changes of the illumination conditions (equation (5.6)). For our database experiments, perceptual variability can result (i) from changes of the illumination during the collection of the image data (`day|day` group), (ii) from changes of the illumination introduced by cross-database experiments (`day|night` group), or (iii) from the image disturbances described in section 5.3.1.1 (both test groups). Based on the dissimilarity values as computed by equation (5.58), perceptual aliasing and perceptual variability cannot be distinguished because in both cases at least one snapshot pair $\boldsymbol{I}_i$ and $\boldsymbol{I}'_j$ exists for which

$$\exists j : \hat{\ell}_{i,j} = \mathrm{d}(\boldsymbol{I}_i, \boldsymbol{I}'_j) \le \hat{\ell}_{i,i} = \mathrm{d}(\boldsymbol{I}_i, \boldsymbol{I}'_i) \text{ with } i \ne j \text{ and } 0 \le i, j < n_x n_y \tag{5.66}$$

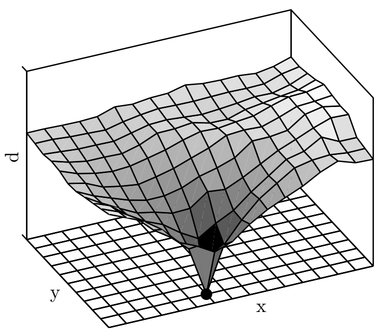
holds. This means that at least one pair of images $\boldsymbol{I}_i$ and $\boldsymbol{I}'_j$ exists for which the resulting dissimilarity $\hat{\ell}_{i,j}$ is smaller than $\hat{\ell}_{i,i} = \mathrm{d}(\boldsymbol{I}_i, \boldsymbol{I}'_i)$ obtained from comparing the image $\boldsymbol{I}_i$ and its disturbed variant $\boldsymbol{I}'_i$. Regarding the dissimilarity profile resulting from plotting the dissimilarities $\hat{\ell}_{i,j}$ vs. the positions of their image acquisition, this means that the *global* minimum is moved away from its expected position i (figure 5.10). *Local* minima may exist, but are not considered for evaluation.

To assess the method's robustness against perceptual aliasing and perceptual variability, we compute the percentage of correct matches

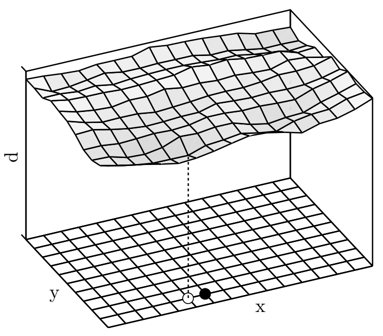
$$P_{\texttt{corr}} = \frac{1}{n_x n_y} \# \left\{ i \;\middle|\; \forall j : \hat{\ell}_{i,j} > \hat{\ell}_{i,i} \text{ with } i \ne j \text{ and } 0 \le i, j < n_x n_y \right\} \tag{5.67}$$

and of mismatches

$$P_{\texttt{mis}} = \frac{1}{n_x n_y} \# \left\{ i \;\middle|\; \exists j : \hat{\ell}_{i,j} < \hat{\ell}_{i,i} \text{ with } i \ne j \text{ and } 0 \le i, j < n_x n_y \right\} \tag{5.68}$$



(1) Ideal case



(2) Disturbed case with perceptual aliasing or perceptual variability

Figure 5.10.: Dissimilarity profile resulting from plotting the pairwise image dissimilarity $\ell_{i,j}$ obtained for fixed i and varying j $(0 \leq j < n_x n_y)$ vs. the position of image acquisition. Subfigure (1) visualizes the ideal case without perceptual aliasing or perceptual variability. The minimum is pronounced and located at the expected position (filled black circle). Subfigure (2) visualizes the occurrence of perceptual aliasing or perceptual variability. In this case, the global minimum (open circle) is moved away from its expected position (filled black circle), the dissimilarities achieve larger values, the optimum is not clearly pronounced, and additional local minima exist. In the depicted case, the distance between the true and expected minima is 1 grid. According to the definition in equation (5.69), the depicted situation would be counted as "close mismatch". In both subfigures, the reference position i was at grid position $(6, 1)$ of the `cliving3night` database; dissimilarities were computed by the $\mathrm{d}_{\mathtt{sad}}$, both figures carry identical axis for d.

among all $n_x n_y$ snapshot positions. For mismatches, the spatial distance between the expected match and the computed match is relevant: mismatches close to the expected match have a smaller influence on the resulting navigation capabilities than mismatches far apart from each other. For the latter, reliable navigation is no longer possible. Therefore, we additionally distinguish between mismatches in the direct vicinity of the expected match

$$P_{\mathtt{misc}} = \frac{1}{n_x n_y} \# \left\{ i \;\middle|\; \text{mismatch and}\, \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq \sqrt{2} g \right\} \tag{5.69}$$

and mismatches further apart

$$P_{\mathtt{misf}} = \frac{100}{n_x n_y} \# \left\{ i \;\middle|\; \text{mismatch and}\, \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) > \sqrt{2} g \right\} \tag{5.70}$$

with $g = 10\,\mathrm{cm}$ being the grid distance of the used image database.

For erroneous matches, we also analyze the spatial distance $\mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ between the true match and the computed match. As performance measures, the mean and median spatial distances between mismatches, $\mathrm{dist}_{\mathrm{avg}}$ and $\mathrm{dist}_{\mathrm{med}}$, are computed over all considered snapshots i for which mismatches occur. In case $\hat{\ell}_{i,j} < \hat{\ell}_{i,i}$ holds for several images $\boldsymbol{I}'_j$, the location x_j at which the image resulting in the smallest value of $\hat{\ell}_{i,j}$ was acquired is selected for the computation of the spatial distance.

With these measures, we can assess the proportion of correct and incorrect loop-closure detections. For incorrect detections, they allow us to draw conclusions on how much navigation is influenced by the misclassifications. The influence of a small number of mismatches in the close vicinity to the expected match will have less influence than a large proportion of distant mismatches which can cause navigation to fail. Results of this evaluation for the standard compass method are described in section 5.4.2; an application to the accelerated compass method is currently not reasonable (sections 5.5.1 and 5.5.4).

5.3.2.3. Compass Accuracy

In addition to the performance of loop-closure detection, we also assess the influence of image preprocessing methods and different dissimilarity functions on the accuracy of the tested compass methods. Each comparison between the images $\boldsymbol{I}_i$ and $\boldsymbol{I}'_j$ not only results in a dissimilarity value $\hat{\ell}_{i,j}$ but also in a shift $\hat{s}_{i,j}$ between the images, which is due to a change of the robot's orientation between image acquisitions. As we used image databases (section 5.3.1.1), the true shift $t_{i,j}$ between the considered images is known, and the absolute deviation

$$e_{i,j} = |t_{i,j} - \hat{s}_{i,j}| \tag{5.71}$$

in pixels can be computed. Due to the horizontal periodicity of the panoramic images, the maximum absolute deviation is $\frac{w}{2}$ with w being the width of the images (with the above definition, $e_{i,j}$ has to be truncated accordingly). The pixel deviation can be converted to an angular error (textbook: [32]) by

$$\delta_{i,j} = e_{i,j} \frac{360°}{w}. \tag{5.72}$$

Angular errors are always in the range $[0°, 180°]$, and to derive scalar measures from the set of computed angular errors, the average angular error (AAE) and the median angular error (MAE) can be computed and compared (textbook: [32]). These measures are also used for evaluating homing methods developed by our group (publications: [447, 451, 452, 454, 660], technical reports: [213, 217, 218]). Since the mean and median angular error are influenced by the spatial distance between the considered snapshots and since we used image databases of different sizes, we only consider image pairs with a spatial distance up to 1 m. We decided for distances up to 1 m because it is the distance typically considered for local visual homing in our trajectory controller (chapter 4) plus some tolerance to take inaccuracies into account. Sections 5.4.3 and 5.5.2 describe the results of evaluating the compass accuracy for the standard and the accelerated compass methods.

5.3.2.4. Computational Aspects

To assess the computing time required for loop-closure detection, we measure the times required to (i) preprocess a single image with a preprocessing function p and (ii) to compute the visual compass method which compares and aligns a pair of preprocessed images (excluding time required for preprocessing). In both cases, the measurements are performed for all possible parameters of preprocessing or dissimilarity functions. To achieve more reliable measurements, images were randomly selected and the median over 1000 repetitions is computed. The measurements rely on the Linux system function `gettimeofday` returning time with a resolution of microseconds (e.g. [63]). The experiments were performed on a desktop computer equipped with an Intel Core i7-870 CPU running at a maximum clock frequency of 2.93 GHz and equipped with 4 GB of RAM. Results are presented in sections 5.4.4, 5.4.5 and 5.5.3.

5.4. Results of Standard Compass Method

This section describes results of the standard compass method for analyzing the loop-closure detection performance (section 5.4.1), the robustness against perceptual aliasing and perceptual variability (section 5.4.2), the compass accuracy (section 5.4.3), the computational effort of image preprocessing (section 5.4.4), and of the visual compass method (section 5.4.5). Conclusions are drawn and discussed in section 5.4.6.

Each combination of preprocessing functions p and dissimilarity functions d is tested with two *test groups*: (i) `day|day` containing a total of approximately 500 000 image pairs collected during the

Table 5.4.: Best combinations of standard compass method w.r.t. loop-closure detection performance. Subtables (1) to (4) contain for each of the four *evaluation* groups (`id-mi`, `id-other`, `pp-mi`, and `pp-other`) the best methods identified based on AUC values of the *test* groups `day|day` and `day|night`. Categories for AUC values are defined in table 5.3 and are given in the form ${}^{\texttt{day|day}}/_{\texttt{day|night}}$.

(1) `id-mi`

Cat.	p	d	AUC values `day|day`	AUC values `day|night`
$^{+++}/_{+}$	$\mathrm{p}_{\texttt{id}}$	$\mathrm{d}_{\texttt{mi3}}$ $(b = 16)$	1.000	0.969
	$\mathrm{p}_{\texttt{id}}$	$\mathrm{d}_{\texttt{mi4}}$ $(b = 32)$	1.000	0.968
	$\mathrm{p}_{\texttt{id}}$	$\mathrm{d}_{\texttt{mi2}}$ $(b = 8)$	1.000	0.965
	$\mathrm{p}_{\texttt{id}}$	$\mathrm{d}_{\texttt{mi5}}$ $(b = 64)$	1.000	0.965
	$\mathrm{p}_{\texttt{id}}$	$\mathrm{d}_{\texttt{mi1}}$ $(b = 4)$	1.000	0.935

(2) `id-other`

Cat.	p	d	AUC values `day|day`	AUC values `day|night`
$^{+++}/_{\circ}$	$\mathrm{p}_{\texttt{id}}$	$\mathrm{d}_{\texttt{zsad}}$	1.000	0.648
	$\mathrm{p}_{\texttt{id}}$	$\mathrm{d}_{\texttt{sad}}$	1.000	0.645
	$\mathrm{p}_{\texttt{id}}$	$\mathrm{d}_{\texttt{ssad}}$	1.000	0.603

(3) `pp-mi`

Cat.	p	d	AUC values `day|day`	AUC values `day|night`
$^{+++}/_{+++}$	$\mathrm{p}_{\texttt{pw}}$	$\mathrm{d}_{\texttt{mi3}}$ $(b = 16)$	1.000	1.000
	$\mathrm{p}_{\texttt{sob}}$	$\mathrm{d}_{\texttt{mi3}}$ $(b = 16)$	1.000	1.000
	$\mathrm{p}_{\texttt{pw}}$	$\mathrm{d}_{\texttt{mi4}}$ $(b = 32)$	1.000	1.000
	$\mathrm{p}_{\texttt{sob}}$	$\mathrm{d}_{\texttt{mi4}}$ $(b = 32)$	1.000	1.000
	$\mathrm{p}_{\texttt{pw}}$	$\mathrm{d}_{\texttt{mi5}}$ $(b = 64)$	1.000	1.000
	$\mathrm{p}_{\texttt{sob}}$	$\mathrm{d}_{\texttt{mi5}}$ $(b = 64)$	1.000	1.000

(4) `pp-other` (both methods with $\sigma_{1/2} = 50$)

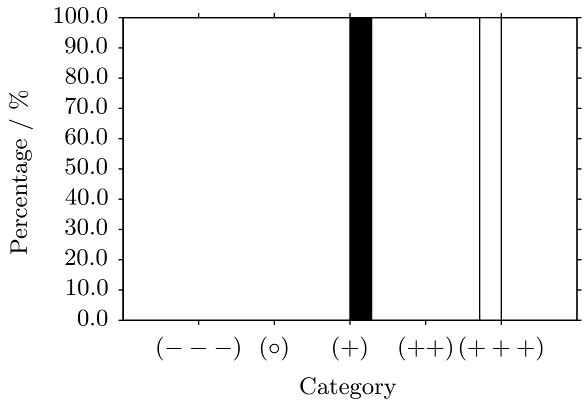
Cat.	p	d	AUC values `day|day`	AUC values `day|night`
$^{+++}/_{++}$	$\mathrm{p}_{\texttt{dv1/4}}$ $(w' = 5)$	$\mathrm{d}_{\texttt{cc}}$	1.000	0.987
	$\mathrm{p}_{\texttt{dv2/4}}$ $(w' = 7)$	$\mathrm{d}_{\texttt{cc}}$	1.000	0.980

day under (nearly) constant illumination conditions, and (ii) `day|night` consisting of approximately 108,000 image pairs taken from cross-databases simulating strong changes of the illumination conditions (section 5.3.1.1). With the parameters of the preprocessing functions p and dissimilarity functions d as defined in section 5.3.1.2, we test a total of 740 methods, i.e. different combinations of p and d.

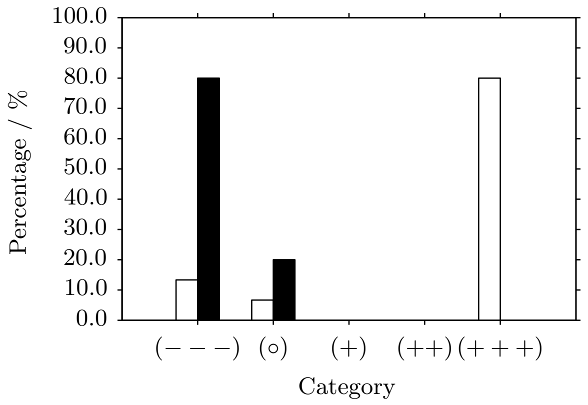
Each combinations is analyzed with three different *evaluation methods* to assess its loop-closure detection performance, its robustness against perceptual aliasing and perceptual variability, and its compass accuracy (section 5.4.3). The evaluation methods yield a single (loop-closure detection performance), two (compass accuracy), or five (robustness against perceptual aliasing and perceptual variability) measures describing the performance of a method. This particular evaluation procedure leads to a huge amount of data which cannot be presented comprehensively in this dissertation. The presentation is therefore restricted to a small subset of the entire data which is still large enough to compare different methods, to draw conclusions, and to answer the objectives outlined at the beginning of section 5.3. We divide the combinations of preprocessing functions and dissimilarity functions into four *evaluation groups*: `id-mi`, `id-other`, `pp-mi`, and `pp-other`. The groups were chosen to assess the influences (i) of using images without prior preprocessing vs. using preprocessed images and (ii) of comparing images by mutual information vs. comparing images by other image dissimilarity functions. For each of the four *evaluation groups* and each of the three *evaluation methods*, we seek for the combination of image preprocessing and dissimilarity functions which achieves the best accuracy for *both* test groups. By this means, a total of twelve combinations is obtained.

5.4.1. Performance of Loop-Closure Detection

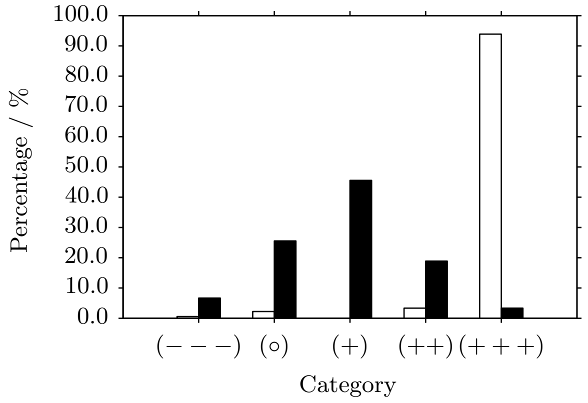
To measure the performance of loop-closure detection, we computed AUC values which were assigned to five bins depending on their AUC values (section 5.3.2.1 and table 5.3). Figure 5.11 depicts for



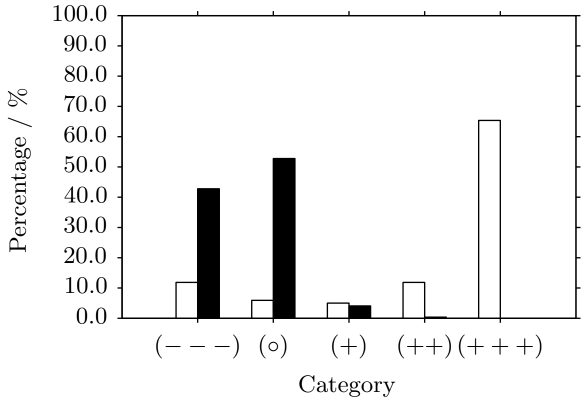
(1) `id-mi`



(2) `id-other`



(3) `pp-mi`



(4) `pp-other`

Figure 5.11.: Frequency distributions of AUC values for the standard compass method. The distributions are used to identify promising methods for loop-closure detection as described in section 5.3.2.1. Subfigures (1) to (4) show the frequency distributions computed for each of the four *evaluation* groups `id-mi`, `id-other`, `pp-mi`, and `pp-other`. White and black bars depict the distributions for the *test* groups `day|day` and `day|night`, respectively. Binning was done w.r.t. the categories defined in table 5.3 as follows: (− − −) loop-closure detection not possible; (○) moderate loop-closure detection performance; (+) good performance; (++) very good performance; (+ + +) ideal results (i.e. AUC = 1.0).

each of the four evaluation groups the frequency distributions for these bins with white and black bars representing the distributions for the `day|day` and `day|night` groups, respectively. The overall impression clearly reveals that the performance for the `day|day` group is considerably better than that of the `day|night` group. For the `day|day` group most methods achieve perfect classifications (+ + +), whereas the results for the `day|night` group are distributed more broadly with a tendency towards worse categories. Based on the histograms depicted in figure 5.11, we identify for each of the four evaluation groups methods which achieve the best performance over both test groups (table 5.4) and which are analyzed in more detail.

All five methods of the evaluation group `id-mi` achieve a perfect classification (+ + +) of the `day|day` group and a good classification (+) for the `day|night` group (figure 5.11.1). With an AUC value for the `day|night` group of 0.969, the best performance is achieved for $b = 16$ histogram bins; for 8, 32 or 64 bins, the performance is only slightly worse (table 5.4.1). The worst performance is obtained for $b = 4$ histogram bins resulting in an AUC value of 0.935. For the `id-other` group, the best possible method can achieve perfect classification (+ + +) for the `day|day` test group and moderate classification results (○) for the `day|night` group (figure 5.11.2). All these methods rely on variants of the sum of absolute differences function and achieve AUC values for the `day|night`

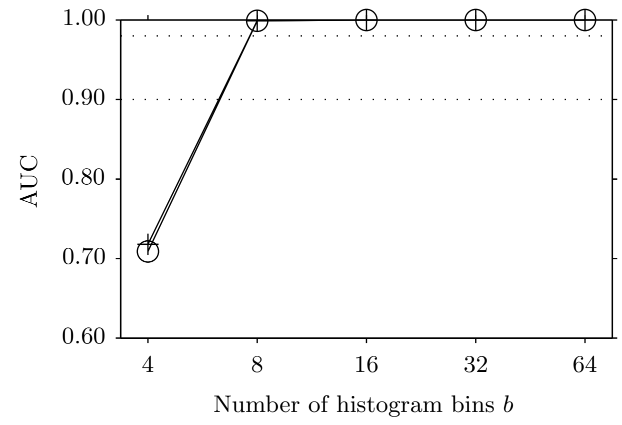


Figure 5.12.: Detailed AUC values for the best methods of the evaluation group `pp-mi`. AUC values are plotted against the number of histogram bins b used for computing the mutual information $\mathrm{d}_{\mathtt{mi}}$. The different lines represent the values obtained for preprocessing with first-order edge detectors $\mathrm{p}_{\mathtt{pw}}$ (∘) and $\mathrm{p}_{\mathtt{sob}}$ (+) Dotted horizontal lines depict the bin limits as defined in table 5.3.

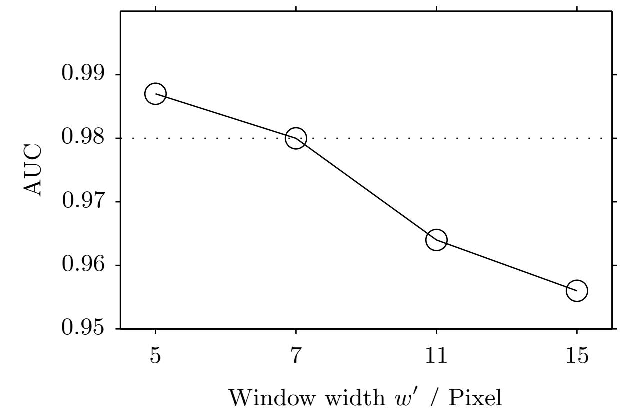


Figure 5.13.: Detailed AUC values for the best methods of the evaluation group `pp-other`. AUC values obtained for comparing images by the cross-correlation function $\mathrm{d}_{\mathtt{cc}}$ are plotted against the window size w' used for the dividing by variation preprocessing $\mathrm{p}_{\mathtt{dv}}$ with $\sigma_{1/2} = 50$. Dotted horizontal lines depict the bin limits as defined in table 5.3.

group between 0.603 and 0.648 (table 5.4.2).

For comparing preprocessed images with mutual information (`pp-mi`; figure 5.11.3 and table 5.4.3) ideal results are achieved for both test groups. Figure 5.12 visualizes the influence of the bin number of histogram bins b used to compute the mutual information $\mathrm{d}_{\mathtt{mi}}$ onto the AUC values. Both first-order edge detectors ($\mathrm{p}_{\mathtt{pw}}$ and $\mathrm{p}_{\mathtt{sob}}$) achieve a perfect classification if more than 16 histogram bins are used for computing the mutual information $\mathrm{d}_{\mathtt{mi}}$ (figure 5.12). For $b = 8$, both methods achieve AUC values close to 1.0; for $b = 4$ the resulting AUC value is approximately 0.7. The performance differences between $\mathrm{p}_{\mathtt{pw}}$ and $\mathrm{p}_{\mathtt{sob}}$ are negligible. The best method of the evaluation group `pp-other` achieves perfect results (+ + +) for the `day|day` test group and very good classification (++) for the `day|night` group (figure 5.11.4 and table 5.4.4). The best methods of this evaluation group all rely on the dividing by variation preprocessing $\mathrm{p}_{\mathtt{dv*/4}}$ with $\sigma_{1/2} = 50$ and compare images with the cross-correlation function $\mathrm{d}_{\mathtt{cc}}$. Figure 5.13 visualizes the AUC values obtained for $\mathrm{p}_{\mathtt{dv*/4}}$ and $\mathrm{d}_{\mathtt{cc}}$ depending on the window size w'. The AUC values decrease for larger windows.

Table 5.5 summarizes the best combinations obtained for the four cases identified in the previous evaluation step. For the `day|day` group, perfect classification with AUC values of 1.0 is achieved for all four cases. The performance for the `day|night` group depends on the combination of preprocessing function p and dissimilarity function d, but is always worse than the performance of the `day|day` test group. Among the combinations listed in table 5.5, the worst performance is obtained for comparing images without prior preprocessing with a dissimilarity function excluding mutual information (i.e. for evaluation group `id-other`). In this case, the AUC value is only 0.648 and was obtained for comparing images by the zero-mean sum of absolute differences function $\mathrm{d}_{\mathtt{zsad}}$. With AUC values of 0.648 and below, we do not consider methods of this class to be applicable for loop-closure detection under strong changes of the illumination. By comparing unprocessed images with the mutual information (`id-mi`) the performance can be increased to an AUC value of

Table 5.5.: Summary of best combinations of standard compass method w.r.t. loop-closure detection performance. The table summarizes for each of the four *evaluation* groups (`id-mi`, `id-other`, `pp-mi`, and `pp-other`) the best methods from table 5.4. AUC values are given for the two *test* groups `day|day` and `day|night`. The bottom row marks whether we consider the best methods of an evaluation group to be applicable (✓) or not (✗) to reliable and accurate loop-closure detection under constant (`day|day`) *and* changing (`day|night`) illumination conditions. We consider a method to be applicable if both AUC values are greater than or equal to 0.9

		Evaluation groups			
		`id-mi`	`id-other`	`pp-mi`	`pp-other`
Best methods	p	$\mathrm{p}_{\mathtt{id}}$	$\mathrm{p}_{\mathtt{id}}$	$\mathrm{p}_{\mathtt{pw}}$, $\mathrm{p}_{\mathtt{sob}}$	$\mathrm{p}_{\mathtt{dv4/4}}$ ($w' = 15$, $\sigma_{1/2} = 50$)
	d	$\mathrm{d}_{\mathtt{mi3}}$	$\mathrm{d}_{\mathtt{zsad}}$	$\mathrm{d}_{\mathtt{mi3}}$, $\mathrm{d}_{\mathtt{mi4}}$, $\mathrm{d}_{\mathtt{mi5}}$ $b \in \{16, 32, 64\}$	$\mathrm{d}_{\mathtt{cc}}$
`day|day`	AUC	1.000	1.000	1.000	1.000
`day|night`	AUC	0.969	0.648	1.000	0.987
Applicability		✓	✗	✓	✓

0.969. This value was obtained for approximating the gray-value distribution by histograms with $b = 16$ bins. The best methods of the evaluation groups `id-mi` and `id-other`, i.e. for comparing images without prior preprocessing, perform worse than the best methods of the groups `pp-mi` and `pp-other`, i.e. for comparing images after prior preprocessing. The second best performance is obtained for comparing images preprocessed by the dividing by variation function $\mathrm{p}_{\mathtt{dv4/4}}$ and computing the image dissimilarity by the cross correlation $\mathrm{d}_{\mathtt{cc}}$. The resulting AUC value is then 0.987 (`pp-other`). The best results are obtained for comparing preprocessed image with the mutual information (`pp-mi`). The first-order edge detectors $\mathrm{p}_{\mathtt{pw}}$ and $\mathrm{p}_{\mathtt{sob}}$ in combination with the mutual information $\mathrm{d}_{\mathtt{mi}}$ with $b \in \{16, 32, 64\}$ achieve a perfect classification of both test groups. With these results, the best methods for the `id-mi`, `pp-mi`, and `pp-other` evaluation groups are applicable for reliable loop-closure detection even if strong changes of the illumination occur.

The overall conclusion is that image preprocessing methods cannot increase the loop-closure detection performance for almost constant illumination (test group `day|day`) because perfect classification is possible in this case. However, image preprocessing does not decrease the performance. For very strong changes as contained in the cross-dataset `day|night`, preprocessing can drastically increase the performance of loop-closure detection. The best methods identified in this section then achieve a perfect or nearly perfect classification. This allows to choose a specific combination of preprocessing function p and image dissimilarity function d depending on the computational complexity of the methods.

5.4.2. Robustness Against Perceptual Aliasing and Perceptual Variability

To assess the method's robustness against perceptual aliasing and perceptual variability, we follow the evaluation procedures as introduced in section 5.3.2.2 and analyze the percentage of correct loop-closure detections $P_{\mathtt{corr}}$, the percentage of mismatches in the direct vicinity of the expected match $P_{\mathtt{misc}}$, and the percentage of distant mismatches $P_{\mathtt{misf}}$. Additionally, we evaluate the mean and median spatial distances $\mathrm{dist}_{\mathrm{avg}}$ and $\mathrm{dist}_{\mathrm{med}}$ between the true match and the erroneous match. These measures do not only assess the classification quality, but also allow to draw conclusions on how strongly navigation is influenced by erroneous loop-closure detections.

The best methods for each of the four evaluation groups are summarized in table 5.6. The overall impression is that for all evaluation groups combinations of preprocessing functions p and image dissimilarity functions d exist which yield ideal results (i.e. 100 % correct classifications) for the `day|day` *test* group. This implies that the measures $P_{\mathtt{misc}}$, $P_{\mathtt{misf}}$, $\mathrm{dist}_{\mathrm{avg}}$, and $\mathrm{dist}_{\mathrm{med}}$ are all

Table 5.6.: Best combinations of standard compass methods w.r.t. robustness against perceptual aliasing and perceptual variability. Subtables (1) to (4) contain for each of the four *evaluation* groups (`id-mi`, `id-other`, `pp-mi`, and `pp-other`) the best methods identified based on the performance measures (section 5.3.2.2) computed for the `day|day` and the `day|night` *test* group. For the `day|day` group, ideal results with 100 % of correct classifications $P_{\texttt{corr}}$ are obtained. We therefore omit the remaining measures which in this case are all equal to zero.

(1) `id-mi`

			Test groups					
			`day|day`	`day|night`				
p	d		$P_{\texttt{corr}}$ / %	$P_{\texttt{corr}}$ / %	$P_{\texttt{misc}}$ / %	$P_{\texttt{misf}}$ / %	dist_{avg}/ cm	dist_{med}/ cm
$p_{\texttt{id}}$	$d_{\texttt{mi4}}$	(b = 64)	100.0	89.7	1.9	8.5	57.0	55.2
$p_{\texttt{id}}$	$d_{\texttt{mi5}}$	(b = 32)	100.0	89.7	2.0	8.3	56.9	55.2
$p_{\texttt{id}}$	$d_{\texttt{mi3}}$	(b = 16)	100.0	88.3	2.2	9.5	56.2	53.9
$p_{\texttt{id}}$	$d_{\texttt{mi2}}$	(b = 8)	100.0	83.1	3.8	13.1	49.8	40.6
$p_{\texttt{id}}$	$d_{\texttt{mi1}}$	(b = 4)	100.0	72.1	6.9	21.0	49.1	31.6

(2) `id-other`

		Test groups					
		`day|day`	`day|night`				
p	d	$P_{\texttt{corr}}$ / %	$P_{\texttt{corr}}$ / %	$P_{\texttt{misc}}$ / %	$P_{\texttt{misf}}$ / %	dist_{avg}/ cm	dist_{med}/ cm
$p_{\texttt{id}}$	$d_{\texttt{zsad}}$	100.0	13.9	7.6	78.5	57.9	50.0
$p_{\texttt{id}}$	$d_{\texttt{sad}}$	100.0	13.1	7.9	79.0	58.2	50.0
$p_{\texttt{id}}$	$d_{\texttt{ssad}}$	100.0	8.8	8.5	82.7	60.7	51.0
$p_{\texttt{id}}$	$d_{\texttt{sssd}}$	100.0	1.8	6.3	91.9	67.0	60.0
$p_{\texttt{id}}$	$d_{\texttt{zssd}}$	100.0	1.6	5.3	93.1	64.2	60.0
$p_{\texttt{id}}$	$d_{\texttt{eucl}}$	100.0	1.4	5.2	93.4	63.7	58.3
$p_{\texttt{id}}$	$d_{\texttt{ssd}}$	100.0	1.4	5.2	93.4	63.7	58.3
$p_{\texttt{id}}$	$d_{\texttt{rms}}$	100.0	1.4	5.2	93.4	63.7	58.3

(3) `pp-mi`

			Test groups					
			`day|day`	`day|night`				
p	d		$P_{\texttt{corr}}$ / %	$P_{\texttt{corr}}$ / %	$P_{\texttt{misc}}$ / %	$P_{\texttt{misf}}$ / %	dist_{avg}/ cm	dist_{med}/ cm
$p_{\texttt{sob}}$	$d_{\texttt{mi3}}$	(b = 16)	100.0	100.0	0.0	0.0	0.0	0.0
$p_{\texttt{sob}}$	$d_{\texttt{mi4}}$	(b = 32)	100.0	100.0	0.0	0.0	0.0	0.0
$p_{\texttt{sob}}$	$d_{\texttt{mi5}}$	(b = 64)	100.0	100.0	0.0	0.0	0.0	0.0
$p_{\texttt{pw}}$	$d_{\texttt{mi3}}$	(b = 16)	100.0	100.0	0.0	0.0	0.0	0.0
$p_{\texttt{pw}}$	$d_{\texttt{mi4}}$	(b = 32)	100.0	100.0	0.0	0.0	0.0	0.0
$p_{\texttt{pw}}$	$d_{\texttt{mi5}}$	(b = 64)	100.0	100.0	0.0	0.0	0.0	0.0

(4) `pp-other`

		Test groups					
		`day|day`	`day|night`				
p	d	$P_{\texttt{corr}}$ / %	$P_{\texttt{corr}}$ / %	$P_{\texttt{misc}}$ / %	$P_{\texttt{misf}}$ / %	dist_{avg}/ cm	dist_{med}/ cm
$p_{\texttt{sob}}$	$d_{\texttt{ssad}}$	100.0	100.00	0.0	0.0	0.0	0.0
$p_{\texttt{pw}}$	$d_{\texttt{ssad}}$	100.0	100.00	0.0	0.0	0.0	0.0

zero. For this reason, table 5.6 does not contain these results for the `day|day` group. Regarding the `day|night` test group, perceptual variability or perceptual aliasing occur only for comparing unprocessed images (*evaluation* groups `id-mi` and `id-other`), but not for comparing preprocessed images (evaluation groups `pp-mi` and `pp-other`). In more details, the results for the four evaluation groups are as follows.

For comparing unprocessed images and computing image dissimilarity by mutual information (`id-mi`), all 5 combinations can perfectly classify the `day|day` group (table 5.6.1). Hence, there are no occurrences of perceptual aliasing or perceptual variability. The performance for the `day|night` group is considerably worse. The optimal performance is obtained for approximating the gray-value distribution by histograms with $b = 32$ or $b = 64$ bins. With these combinations, perceptual aliasing or perceptual variability only occur in approximately 10 % of the cases; 2 % are close mismatches, and with 8 % most of the failed detections are far mismatches. The median and mean spatial distances between the mismatches of $\mathrm{dist}_{\mathrm{med}} = 55.2\,\mathrm{cm}$ and approximately $\mathrm{dist}_{\mathrm{med}} = 57.0\,\mathrm{cm}$. Thus, reliable loop-closure detection under varying illumination conditions is not possible.

Table 5.6.2 shows the best results obtained for the evaluation group `id-other`, i.e. for comparing unprocessed images with dissimilarity functions other than mutual information. For the `day|day` test group, several combinations achieve ideal values. The best methods for the `day|night` group correctly detect loop closures in only 13 % of the cases. In case perceptual aliasing or perceptual variability occur, approximately 7 % are mismatches are not in the direct vicinity of the expected match, but approximately 80 % are far mismatches. The median and mean spatial distances are both larger than 50 cm. These results are obtained for the methods comparing images with the sum of absolute differences function $\mathrm{d}_{\mathtt{sad}}$ and the zero-mean variant $\mathrm{d}_{\mathtt{zsad}}$. Similar to the evaluation group `id-mi`, reliable and accurate loop closure detection is only under nearly constant illumination conditions, but not under the strong changes of illumination contained in the `day|night` group.

For the evaluation group `pp-mi` (table 5.6.3), the best results are obtained by the combinations of the first-order edge detectors $\mathrm{p}_{\mathtt{pw}}$ and $\mathrm{p}_{\mathtt{sob}}$ with mutual information with more than 16 bins. All these six combinations achieve ideal results for the `day|day` and the `day|night` groups. If mutual information is excluded (`pp-other`), the first-order edge detectors in combination with the scaled sum of squared differences function $\mathrm{d}_{\mathtt{sssd}}$ also achieve ideal results for both test groups (table 5.6.4).

Table 5.7 summarizes the best combinations identified by analyzing the robustness against perceptual aliasing and perceptual variability. In all four *evaluation* groups, ideal results are obtained for the `day|day` *test* group. Performance differences are only found for the `day|night` group. For comparing images without prior preprocessing (evaluation groups `id-mi` and `id-other`), the performance strongly decreases: a substantial proportion of loop closures is not correctly detected, and the mean and median distances between expected and found matches are larger than 50 cm. Thus, these methods are only applicable to loop-closure detection under (nearly) constant illumination conditions but fail if strong changes of the illumination occur. The best methods of the two evaluation groups preprocessing images prior to image comparison (`pp-mi` and `pp-other`) achieve ideal results for both test groups. They are therefore applicable to loop-closure detection even under the presence of strong illumination changes. In both cases, the methods preprocessing images by the Prewitt or the Sobel filter ($\mathrm{p}_{\mathtt{pw}}$ and $\mathrm{p}_{\mathtt{sob}}$) and comparing images by mutual information $\mathrm{d}_{\mathtt{mi}}$ with more than 16 histogram bins or the scaled sum of absolute differences $\mathrm{d}_{\mathtt{ssad}}$ perform best.

5.4.3. Compass Accuracy

For assessing the accuracy of the proposed compass methods, we followed the approach described in section 5.3.2.3 and computed the average angular error AAE and the median angular error MAE. Similar to sections 5.4.1 to 5.4.2, we analyzed the compass accuracy for four *evaluation* groups, namely `id-mi`, `id-other`, `pp-mi`, and `pp-other`.

For comparing unprocessed images by mutual information (evaluation group `id-mi`; table 5.8.1),

Table 5.7.: Summary of best combinations of standard compass method w.r.t. robustness against perceptual aliasing and perceptual variability. The table summarizes for each of the four *evaluation* groups (`id-mi`, `id-other`, `pp-mi`, and `pp-other`) the best methods from table 5.6. Performance measures are given for the two *test* groups `day|day` and `day|night`. The bottom row marks whether we consider the method of an evaluation group to be applicable (✓) or not (✗) to loop-closure detection *both* under constant (`day|day`) and changing illumination (`day|night`) conditions. We consider a method to be applicable (✓), if $P_{\texttt{corr}} > 90\,\%$ and $\text{dist}_{\text{med}} < 15\,\text{cm}$.

		Evaluation groups			
		`id-mi`	`id-other`	`pp-mi`	`pp-other`
Best methods	p	$\text{p}_{\texttt{id}}$	$\text{p}_{\texttt{id}}$	$\text{p}_{\texttt{pw}}$, $\text{p}_{\texttt{sob}}$	$\text{p}_{\texttt{pw}}$, $\text{p}_{\texttt{sob}}$
	d	$\text{d}_{\texttt{mi4}}$ ($b = 32$)	$\text{d}_{\texttt{zsad}}$	$\text{d}_{\texttt{mi3}}$, $\text{d}_{\texttt{mi4}}$, $\text{d}_{\texttt{mi5}}$ ($b \in \{16, 32, 64\}$)	$\text{d}_{\texttt{ssad}}$
`day|day`	$P_{\texttt{corr}}$ / %	100.0	100.0	100.0	100.0
	$P_{\texttt{misc}}$ / %	0.0	0.0	0.0	0.0
	dist_{med} / cm	0.0	0.0	0.0	0.0
	dist_{avg} / cm	0.0	0.0	0.0	0.0
`day|night`	$P_{\texttt{corr}}$ / %	89.7	13.9	100.0	100.0
	$P_{\texttt{misc}}$ / %	1.9	7.6	0.0	0.0
	dist_{med} / cm	55.2	50.0	0.0	0.0
	dist_{avg} / cm	57.0	57.9	0.0	0.0
Applicability		✗	✗	✓	✓

the best results for the `day|day` *test* group are obtained for using $b = 64$ histogram bins. For this combination, the median and mean angular errors are 5.47° and 9.41°, respectively. For less histogram bins, the mean angular error increases whereas the median angular error is almost constant. The results obtained for the `day|night` group are much worse with the best values of MAE and AAE being 26.55° and 41.30° being obtained by the mutual information with $b = 32$.

In case images are compared by dissimilarities other than mutual information (`id-other`; table 5.8.2), the image dissimilarity functions $\text{d}_{\texttt{sad}}$, $\text{d}_{\texttt{zsad}}$, and $\text{d}_{\texttt{ssad}}$ for the `day|day` group all obtain a median angular error of 5.47°. The angular errors range between 9.88° and 11.94°. The accuracy for the `day|night` group is much worse: the median and mean angular errors of the best method —in this case for comparing images by the sum of absolute differences $\text{d}_{\texttt{sad}}$— are 43.73° and 65.31°. The performances for comparing images by $\text{d}_{\texttt{zsad}}$ and $\text{d}_{\texttt{ssad}}$ are slightly worse.

For comparing preprocessed images by mutual information, the optimal performance for the `day|day` and `day|night` group are obtained for different combinations (`pp-mi`; table 5.8.3). For the `day|day` group, the best combination is preprocessing images by the difference of Gaussians $\text{p}_{\texttt{dog3}}$ and using $b = 64$ histogram bins for approximating the gray-value distribution. The median and mean angular errors are then 4.69° and 9.70° for the `day|day` group and 10.15° and 23.65° for the `day|night` group. To analyze the influence of the number of histogram bins b on the compass accuracy, figure 5.14 visualizes the AAE vs. the MAE with different markers coding b as explained in the figure's caption. It reveals that —in addition to the methods achieving the best performance— the combination of $\text{p}_{\texttt{dog3}}$ and $\text{d}_{\texttt{mi2}}$ with $b = 8$ histogram bins (□) yields a MAE of 5.47° and an AAE of 8.37° for the `day|day` group. However, its performance for the `day|night` group is rather low. For the `day|night` group, the best performance is obtained by the local contrast $\text{p}_{\texttt{dl4}}$ with a patch size of $w' = 15$ pixels achieving a MAE of 8.59° and an AAE of 16.13°. For the `day|day` group, the combination yields a MAE of 5.47° and an AAE of 9.17°. Figure 5.15 visualizes the accuracy of $\text{p}_{\texttt{dl4}}$ depending on the number of histogram bins b used for computing the mutual information. It clearly shows that the accuracy is better for large choices of b and worse for smaller values of b. In both cases the AAE grows stronger than the MAE therefore suggesting the existence of a larger number of outliers.

Table 5.8.: Best combinations of standard compass method w.r.t. compass accuracy. Subtables (1) to (4) contain for each of the four *evaluation* groups (`id-mi`, `id-other`, `pp-mi`, and `pp-other`) the median angular error (MAE) and the average angular error (AAE) computed for the `day|day` and `day|night` *test* groups. The highlighted columns mark the measure used for sorting. In case several combinations achieve identical MAE values, the AAE was used as secondary sorting criterion. For the evaluation groups `pp-mi` and `pp-other` (subtables (3) and (4), different methods achieve the best performance for the `day|day` or the `day|night` groups. In this case, the upper part of each subtable contains the best methods identified for the `day|day`test group, and the lower part contains the best methods identified for the `day|day` group.

(1) `id-mi`

			Test groups			
			`day|day`		`day|night`	
p	d		MAE / °	AAE / °	MAE / °	AAE / °
$p_{\texttt{id}}$	$d_{\texttt{mi5}}$	$(b = 64)$	5.47	9.41	28.11	42.77
$p_{\texttt{id}}$	$d_{\texttt{mi4}}$	$(b = 32)$	5.47	9.61	26.55	41.30
$p_{\texttt{id}}$	$d_{\texttt{mi3}}$	$(b = 16)$	5.47	9.68	28.89	42.57
$p_{\texttt{id}}$	$d_{\texttt{mi2}}$	$(b = 8)$	5.47	10.31	35.14	47.94
$p_{\texttt{id}}$	$d_{\texttt{mi1}}$	$(b = 4)$	6.25	12.05	37.48	52.40

(2) `id-other`

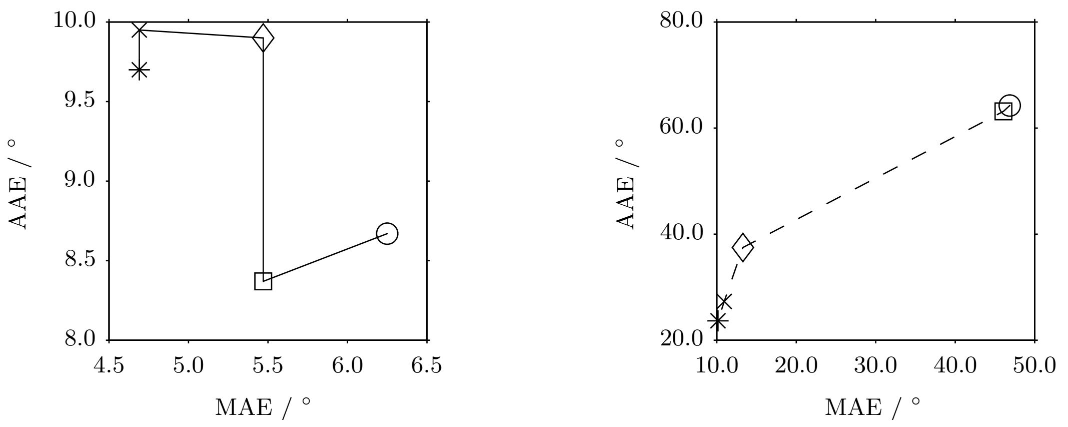
		Test groups			
		`day|day`		`day|night`	
p	d	MAE / °	AAE / °	MAE / °	AAE / °
$p_{\texttt{id}}$	$d_{\texttt{sad}}$	5.47	9.88	43.73	65.31
$p_{\texttt{id}}$	$d_{\texttt{zsad}}$	5.47	11.91	44.51	63.98
$p_{\texttt{id}}$	$d_{\texttt{ssad}}$	5.47	11.94	44.51	63.75

(3) `pp-mi`

				Test groups			
				`day|day`		`day|night`	
p		d		MAE / °	AAE / °	MAE / °	AAE / °
$p_{\texttt{dog3}}$	$(\sigma_1 = 0.01, \sigma_2 = 0.08)$	$d_{\texttt{mi5}}$	$(b = 64)$	4.69	9.70	10.15	23.65
$p_{\texttt{dog3}}$	$(\sigma_1 = 0.01, \sigma_2 = 0.08)$	$d_{\texttt{mi4}}$	$(b = 32)$	4.69	9.95	10.93	27.32
$p_{\texttt{dl4}}$	$(w' = 15)$	$d_{\texttt{mi5}}$	$(b = 64)$	5.47	9.17	8.59	16.13
$p_{\texttt{dl3}}$	$(w' = 11)$	$d_{\texttt{mi5}}$	$(b = 64)$	5.47	9.20	8.59	16.61
$p_{\texttt{dl4}}$	$(w' = 15)$	$d_{\texttt{mi4}}$	$(b = 32)$	5.47	8.87	8.59	17.22
$p_{\texttt{dl3}}$	$(w' = 11)$	$d_{\texttt{mi4}}$	$(b = 32)$	5.47	9.02	8.59	17.58

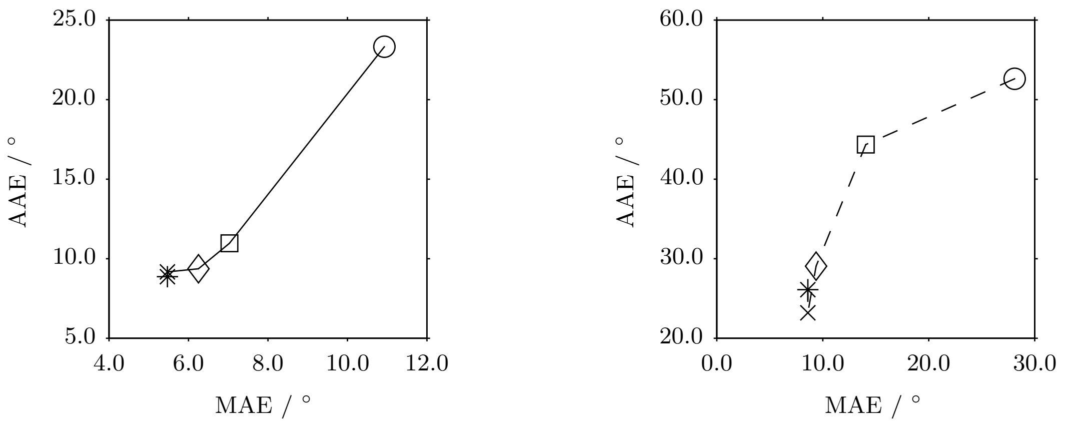
(4) `pp-other`

			Test groups			
			`day|day`		`day|night`	
p		d	MAE / °	AAE / °	MAE / °	AAE / °
$p_{\texttt{heq5}}$	$(b = 64)$	$d_{\texttt{sad}}$	4.69	9.97	42.17	52.53
$p_{\texttt{heq4}}$	$(b = 32)$	$d_{\texttt{sad}}$	4.69	9.98	42.17	52.57
$p_{\texttt{heq5}}$	$(b = 64)$	$d_{\texttt{zsad}}$	4.69	10.00	42.17	52.56
$p_{\texttt{heq5}}$	$(b = 64)$	$d_{\texttt{ssad}}$	4.69	10.00	42.17	52.57
$p_{\texttt{heq4}}$	$(b = 32)$	$d_{\texttt{zsad}}$	4.69	10.03	42.17	52.60
$p_{\texttt{heq4}}$	$(b = 32)$	$d_{\texttt{ssad}}$	4.69	10.03	42.17	52.61
$p_{\texttt{heq3}}$	$(b = 16)$	$d_{\texttt{sad}}$	4.69	10.04	42.17	52.63
$p_{\texttt{heq3}}$	$(b = 16)$	$d_{\texttt{zsad}}$	4.69	10.09	42.17	52.66
$p_{\texttt{heq3}}$	$(b = 16)$	$d_{\texttt{ssad}}$	4.69	10.10	42.17	52.68
$p_{\texttt{dl2}}$	$(w' = 7)$	$d_{\texttt{sad}}$	5.47	10.46	9.37	18.18



(1) `day|day` group **(2)** `day|night` group

Figure 5.14.: Compass accuracy of standard compass method for preprocessing by difference of Gaussians $\mathrm{p}_{\mathtt{dog3}}$ and comparing images by mutual information $\mathrm{d}_{\mathtt{mi}}$. Subfigures (1) and (2) depict the results for the two *test* groups `day|day` and `day|night`, respectively. Each subfigure visualizes the AAE vs. the MAE depending on the number of histogram bins b used for computing $\mathrm{d}_{\mathtt{mi}}$. The number of histogram bins is coded as follows: ($\circ$) $b = 4$, ($\square$) $b = 8$, ($\diamond$) $b = 16$, ($\times$) $b = 32$ and ($*$) $b = 64$. The parameters of $\mathrm{p}_{\mathtt{dog3}}$ are $\sigma_1 = 0.01$ and $\sigma_2 = 0.08$.



(1) `day|day` group **(2)** `day|night` group

Figure 5.15.: Compass accuracy of standard compass method for preprocessing by local contrast $\mathrm{p}_{\mathtt{dl4}}$ and comparing images by mutual information $\mathrm{d}_{\mathtt{mi}}$. Subfigures (1) and (2) depict the results for the two *test* groups `day|day` and `day|night`, respectively. The subfigures visualize the AAE vs. the MAE depending on the number of histogram bins b used for computing $\mathrm{d}_{\mathtt{mi}}$. The number of histogram bins is coded as follows: ($\circ$) $b = 4$, ($\square$) $b = 8$, ($\diamond$) $b = 16$, ($\times$) $b = 32$ and ($*$) $b = 64$. The patch size for the preprocessing is $w' = 15$.

For the remaining evaluation group `pp-other`, i.e. comparing preprocessed images with dissimilarity functions except mutual information, different combinations yield optimal performance depending on the measure used for ranking (table 5.8.4). For ranking depending on the MAE vales of the `day|day` group, the optimal combination is histogram equalization $\mathrm{p}_{\mathtt{heq4}}$ with $b = 64$ bins and comparing images by the sum of absolute differences. This combination results in a MAE of 4.69° and an AAE of 9.97°. For the `day|night` group, the resulting accuracy is strongly decreased and only achieves a MAE of 42.17° and an AAE of 52.53°. In both cases, histograms with a larger number of histogram bins b yield a better compass accuracy (figure 5.16). For the `day|night` group, the best combination is image preprocessing by the local contrast $\mathrm{p}_{\mathtt{dl2}}$ with a patch size of $w' = 7$ pixels and comparing images by the sum of absolute differences. The resulting MAE and AAE values are 9.37° and 18.18° for the `day|night` group and 5.47° and 10.46° for the `day|day` group.

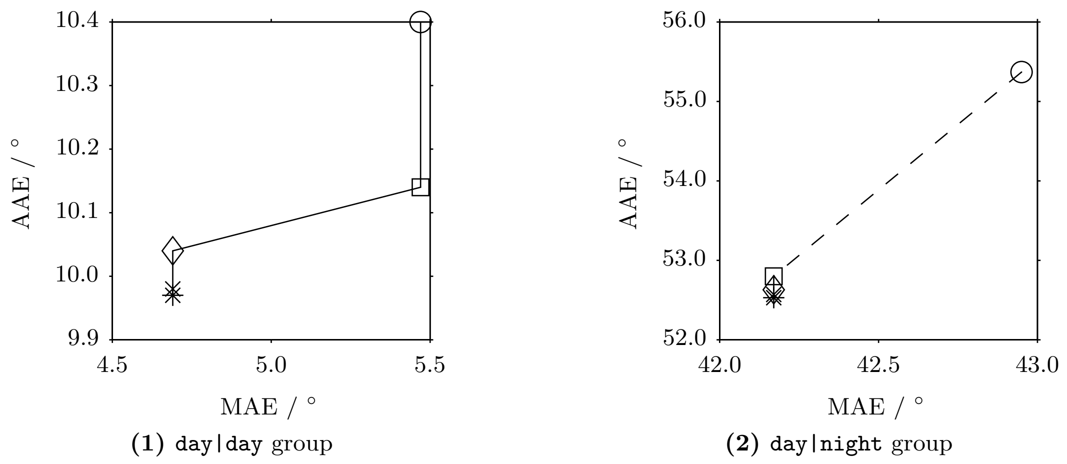


Figure 5.16.: Compass accuracy of standard compass method for preprocessing by histogram equalization $p_{\mathtt{heq}}$ and comparing images by the sum of absolute differences $d_{\mathtt{sad}}$. Subfigures (1) and (2) show the results for the `day` and `day|night` test groups. Each subfigure depicts the AAE vs. the MAE depending on the number of histogram bins b used for computing the histogram equalization. The number of histogram bins b is coded as follows: (○) $b = 4$, (□) $b = 8$, (◇) $b = 16$, (×) $b = 32$ and (∗) $b = 64$.

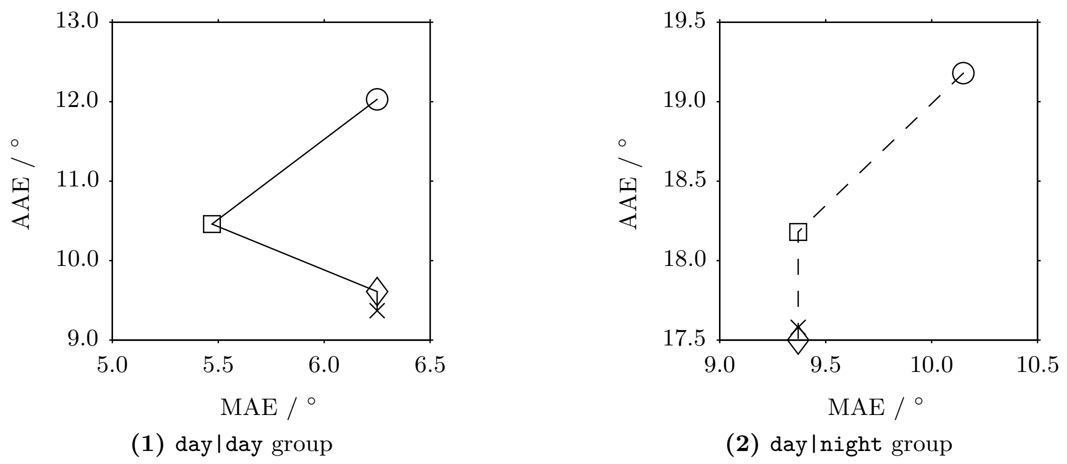


Figure 5.17.: Compass accuracy of standard compass method for preprocessing by local contrast $p_{\mathtt{dl}}$ and comparing images by the sum of absolute differences $d_{\mathtt{sad}}$. Subfigures (1) and (2) show the results for the `day` and `day|night` test groups. Each subfigure visualizes the AAE vs. the MAE depending on the patch size w' used by $p_{\mathtt{dl}}$. Patch sizes are coded as follows: (○) $b = 4$, (□) $b = 8$, (◇) $b = 16$ and (×) $b = 32$.

Thus the compass accuracy of that combination for the `day|day` group is only slightly worse than the accuracy of the optimal combination. The average angular errors obtained with the combination are smaller for larger patch sizes w' (figure 5.17).

Table 5.9 summarizes the results obtained for analyzing the method's compass accuracy. In case different combinations performed best for the `day|day` and the `day|night` test groups (tables 5.6.3 and 5.6.4), the optimal combination for the `day|night` group is given in table 5.9. These combinations perform only slightly worse for the `day|day` group, but much better for the `day|night` group (tables 5.8.3 and 5.8.4). For the `day|day` test group, the median angular error MAE of all four considered evaluation groups is 5.47°; the corresponding average angular errors (AAE) range between 9.17° and 10.46°. Thus, the accuracy differences of the best methods for the `day|day` group are rather small. The resulting accuracy for the `day|night` group is in all four cases worse than for the `day|day` group. The median angular errors of the `day|night` group range from 8.59° to

Table 5.9.: Summary of best combinations of standard compass method w.r.t. compass accuracy. The table summarizes for each of the four *evaluation* groups (`id-mi`, `id-other`, `pp-mi`, and `pp-other`) the best methods from table 5.8. Mean (AAE) and median (MAE) angular errors are given for the two *test* groups `day|day` and `day|night`. The bottom row marks whether we consider the best method of a group to be applicable (✓) or not (✗) as a compass under constant and varying illumination conditions. We consider a method to be applicable if the MAE values of both test groups are equal to or less than 10°.

		Evaluation groups			
		`id-mi`	`id-other`	`pp-mi`	`pp-other`
Best methods	p	$\mathrm{p}_{\texttt{id}}$	$\mathrm{p}_{\texttt{id}}$	$\mathrm{p}_{\texttt{dl4}}$ ($w' = 15$)	$\mathrm{p}_{\texttt{dl2}}$ ($w' = 7$)
	d	$\mathrm{d}_{\texttt{mi5}}$ ($b = 64$)	$\mathrm{d}_{\texttt{zsad}}$	$\mathrm{d}_{\texttt{mi5}}$ ($b = 64$)	$\mathrm{d}_{\texttt{sad}}$
`day|day`	MAE / °	5.47	5.47	5.47	5.47
	AAE / °	9.41	9.88	9.17	10.46
`day|night`	MAE / °	28.11	43.73	8.59	9.37
	AAE / °	42.77	65.31	16.13	18.18
Applicability		✗	✗	✓	✓

43.73°, and the average angular errors vary between 16.13° and 65.31°. Computing the change of the robot's orientation based on preprocessed images (`pp-mi`, `pp-other`) gives significantly better results than comparing images without prior preprocessing (`id-mi`, `id-other`). Using the mutual information to compute the image dissimilarity (`id-mi`, `pp-mi`) yields better results than comparing the images by other dissimilarity functions (`id-other`, `pp-other`). For comparing images without prior preprocessing (`id-mi`), mutual information can greatly improve the accuracy, whereas there is only a small improvement for comparing images by $\mathrm{d}_{\texttt{mi}}$ after prior preprocessing (`pp-mi`). Regarding the applicability as a compass method, the combinations without prior image preprocessing (`id-mi` and `id-other`) are only applicable under nearly constant illumination conditions. Only the methods comparing preprocessed images (`pp-mi` and `pp-other`) compute reliable estimates even if strong illumination changes occur.

From these results we conclude that, for small or moderate changes of the illumination, all four considered evaluation groups achieve a good compass accuracy. In case strong changes of the illumination are expected, image preprocessing has a positive influence on the resulting compass accuracy. With respect to image dissimilarity functions, the best results are obtained for comparing images by the mutual information $\mathrm{d}_{\texttt{mi}}$. However, combinations of preprocessing functions p and dissimilarity functions d exist which only yield a slightly worse performance.

5.4.4. Computational Aspects of Image Preprocessing Functions

The results of the time measurements for the tested image preprocessing functions are visualized in figure 5.18; a tabular presentation of the results is given in table C.1. With a computing time of 0.38 ms, the Laplacian $\mathrm{p}_{\texttt{lap}}$ performs best (table C.1.1). Histogram equalization $\mathrm{p}_{\texttt{heq}}$ takes 1.6 ms and is independent of the number of bins b (table C.1.6). The first-order edge detectors $\mathrm{p}_{\texttt{pw}}$ and $\mathrm{p}_{\texttt{sob}}$ both require approximately 7 ms (table C.1.1). Preprocessing the image with a difference of Gaussians $\mathrm{p}_{\texttt{dog}}$ requires approximately 9 ms (table C.1.2). As the convolution is computed in the frequency domain, the computing times are independent of the choice of the parameters σ_1 and σ_2. This is in contrast to the early-vision functions, which —with the current implementation— convolve the image in the spatial domain. Thus, these functions require considerably more effort to compute, and the computing times depend on patch size w'. In the average over all tested parameters, the functions $\mathrm{p}_{\texttt{dl}}$, $\mathrm{p}_{\texttt{dlc}}$, and $\mathrm{p}_{\texttt{dv}}$ require computing times of 84.6 ms, 91.6 ms, and

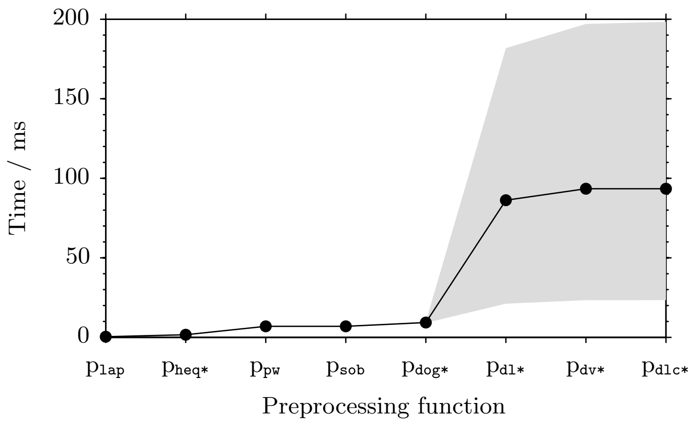


Figure 5.18.: Computing times for image preprocessing methods. For the functions $\mathtt{p_{heq}}$, $\mathtt{p_{dog}}$, $\mathtt{p_{dl}}$, $\mathtt{p_{dlc}}$, and $\mathtt{p_{dv}}$ tested with several parameters, the average computing time is plotted; the shaded area marks the range between the minimum and maximum computing time.

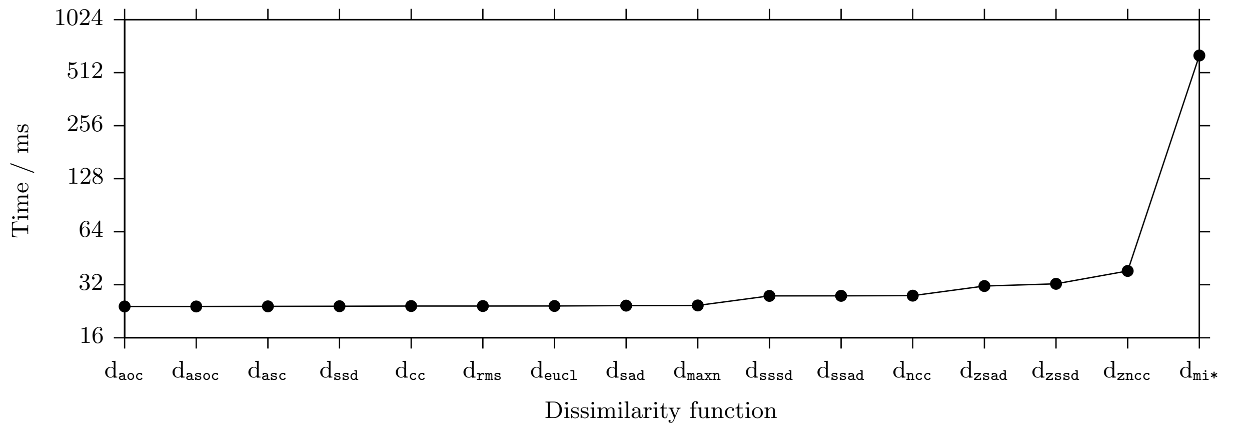


Figure 5.19.: Computing times for global image comparisons. For the mutual information $\mathtt{d_{mi}}$ the average value over the tested numbers of bins $b \in \{4, 8, 16, 32, 64\}$ (equation (5.59)) is given. Please note the logarithmic scale of the vertical axis.

91.8 ms, respectively (tables C.1.3 to C.1.5). Depending on the patch size w', the functions $\mathtt{p_{dlc}}$ and $\mathtt{p_{dv}}$ require approximately 22 ms for $w' = 5$, 43 ms for $w' = 7$, 106 ms for $w' = 11$ ms, and 194 ms for $w' = 15$. Like for the average values, the local contrast $\mathtt{p_{dl}}$ performs slightly better (21 ms for $w' = 5$, 40 ms for $w' = 7$, 99 ms for $w' = 11$, and 178 ms for $w' = 15$). By reimplementing these functions with a convolution in the Fourier domain, a performance comparable or only slightly worse than the difference of Gaussians $\mathtt{p_{dog}}$ should be achievable.

We conclude that preprocessing by applying the Laplacian $\mathtt{p_{lap}}$ and by applying histogram equalization $\mathtt{p_{heq}}$ perform best. The other edge detectors $\mathtt{p_{pw}}$, $\mathtt{p_{sob}}$, and $\mathtt{p_{dog}}$ all require less than 10 ms. This is also the expected computing time the early-vision functions $\mathtt{p_{dl}}$, $\mathtt{p_{dlc}}$, and $\mathtt{p_{dv}}$ would require if the involved convolutions were computed in the frequency domain. We therefore assume a computing time of 10 ms for preprocessing an image as an upper limit for the discussion whether or not the proposed methods are applicable on a real cleaning robot (section 5.6.2).

5.4.5. Computational Aspects of Global Image Dissimilarity Functions

The results obtained for measuring the computing times of the compass function, i.e. determining the compass shift $s_{\min}$ and the image dissimilarity $\ell_{\min}$ according to equations (5.2) and (5.4), are shown in figure 5.19 and in tabular form in table C.2. In the figure, the dissimilarity functions are ranked by increasing computing times. The dissimilarity functions relying on absolute or squared pixel differences without a scaling and without subtracting the average image brightness ($\mathtt{d_{sad}}$, $\mathtt{d_{maxn}}$, $\mathtt{d_{sad}}$, $\mathtt{d_{ssd}}$, $\mathtt{d_{eucl}}$, $\mathtt{d_{rms}}$, $\mathtt{d_{asc}}$, $\mathtt{d_{aoc}}$, and $\mathtt{d_{asoc}}$) as well ass the cross-correlation function $\mathtt{d_{cc}}$

require the least computational effort: they can all be computed within 24 ms. The functions $d_{\mathtt{ssad}}$, $d_{\mathtt{sssd}}$, and $d_{\mathtt{ncc}}$ involving a scaling or a normalization step require 27 ms. With computing times of approximately 32 ms, the functions $d_{\mathtt{zsad}}$ and $d_{\mathtt{zssd}}$ incorporating a subtraction of the average image brightness require slightly more computational effort. The zero-mean normalized cross correlation $d_{\mathtt{zncc}}$ takes 38 ms to compute. As it involves both a normalization and a subtraction of the image brightness, it requires more effort than dissimilarity functions incorporating either a scaling or a subtraction of the average image intensity. By far the largest computational effort is necessary to compute the mutual information $d_{\mathtt{mi}}$. Depending on the number of bins b used to approximate the image's gray-value distribution, it takes between 633 ms and 659 ms.

Thus, most of the comparison functions compute the compass shift and the image dissimilarity in less than 30 ms. We therefore use 30 ms as an estimate for the required computing time in order to draw conclusions about the applicability of the proposed methods in real-robot experiments (section 5.6.2). With computing times of more than 600 ms, mutual information will not be applicable on a real robot. We do not expect that the implementation of the mutual information can be optimized to achieve a performance comparable to that of the other tested dissimilarity functions. For possible optimization approaches beyond an implementation in integer arithmetic, the reader is referred to [286, 516, 617, 627].

5.4.6. Discussion and Conclusions

In the previous evaluation steps (sections 5.4.1 to 5.4.5) we evaluated the accuracy of loop-closure detection, of the visual compass, and the computational efficiency. For the evaluation of the loop-closure detection accuracy (section 5.4.1), the robustness against perceptual aliasing (section 5.4.2), and the compass accuracy (section 5.4.3), the results were categorized into the four *evaluation groups* `id-mi`, `id-other`, `pp-mi`, and `pp-other`. These groups subsume methods, i.e. combinations of preprocessing functions p and dissimilarity functions d depending on whether images were compared with or without prior preprocessing, or whether images were compared by mutual information or other dissimilarity functions. For each of the evaluation groups, the combinations of p and d resulting in the best performance were identified based on tables 5.5, 5.7 and 5.9; the results are summarized in table 5.10.

In the following, the results of the previous evaluation steps (sections 5.4.1 to 5.4.5) will be briefly summarized for each of the four evaluation groups:

`id-mi` The best performance for *loop-closure detection* is achieved by mutual information $d_{\mathtt{mi}}$ with $b = 16$ or $b = 32$ histogram bins for approximating the image's gray-value distribution. Regarding the *compass accuracy*, larger choices of b achieve better results. All the results for the `day|night` test group are considerably worse than the results obtained for the `day|day` group, making the method neither applicable for loop-closure detection nor as a visual compass if strong illumination changes are likely to occur. At least with the current implementation, mutual information is not applicable on a real robot.

`id-other` The best performance for both *loop-closure detection* and as a *visual compass* is obtained for comparing images by the zero-mean sum of squared differences $d_{\mathtt{zsad}}$. While the method is applicable under (nearly) constant illumination conditions, it fails under strong changes of the illumination. Regarding computational efficiency, it is among the most efficient methods considered in this chapter.

`pp-mi` For *loop-closure detection*, first-order edge detection $p_{\mathtt{pw}}$ or $p_{\mathtt{sob}}$ in combination with the mutual information $d_{\mathtt{mi}}$ with $b = 16$ or more bins resulted in the best performance. The method is capable of perfectly detecting loop closures with the `day|day` and `day|night` test groups. In case the proposed methods should be used as *visual compass*, the local

Table 5.10.: Summary of best combinations for standard compass method. The table contains for each of the four *evaluation groups* (`id-mi`, `id-other`, `pp-mi`, `pp-other`) and the three *evaluation methods* (LCDP, RPA, and CA) the combination of preprocessing method p and dissimilarity function d achieving the best performance. The acronyms of the evaluation methods are as follows: LCDP: loop-closure detection performance (section 5.3.2.1), RPA: robustness against perceptual aliasing and perceptual variability (section 5.3.2.2), and CA: compass accuracy (section 5.3.2.3). The data is summarized from tables 5.5, 5.7 and 5.9. Results for the `day|day` and the `day|night` *test groups* are given in the form $^{\mathtt{day|day}}/_{\mathtt{day|night}}$.

			Evaluation groups			
			`id-mi`	`id-other`	`pp-mi`	`pp-other`
Evaluation methods	LCDP	p	$\mathrm{p}_{\mathtt{id}}$	$\mathrm{p}_{\mathtt{id}}$	$\mathrm{p}_{\mathtt{pw}}$, $\mathrm{p}_{\mathtt{sob}}$	$\mathrm{p}_{\mathtt{dv4/4}}$
		d	$\mathrm{d}_{\mathtt{mi3}}$	$\mathrm{d}_{\mathtt{zsad}}$	$\mathrm{d}_{\mathtt{mi3}}$, $\mathrm{d}_{\mathtt{mi4}}$, $\mathrm{d}_{\mathtt{mi5}}$	$\mathrm{d}_{\mathtt{cc}}$
		AUC	1.000 / 0.969	1.000 / 0.648	1.000 / 1.000	1.000 / 0.987
	RPA	p	$\mathrm{p}_{\mathtt{id}}$	$\mathrm{p}_{\mathtt{id}}$	$\mathrm{p}_{\mathtt{pw}}$, $\mathrm{p}_{\mathtt{sob}}$	$\mathrm{p}_{\mathtt{pw}}$, $\mathrm{p}_{\mathtt{sob}}$
		d	$\mathrm{d}_{\mathtt{mi5}}$	$\mathrm{d}_{\mathtt{zsad}}$	$\mathrm{d}_{\mathtt{mi5}}$	$\mathrm{d}_{\mathtt{sad}}$
		P_{corr}/%	100.0 / 89.7	100.0 / 13.9	100.0 / 100.0	100.0 / 100.0
		$\mathrm{dist}_{\mathrm{med}}$ / cm	0.0 / 55.2	0.0 / 50.0	0.0 / 0.0	0.0 / 0.0
	CA	p	$\mathrm{p}_{\mathtt{id}}$	$\mathrm{p}_{\mathtt{id}}$	$\mathrm{p}_{\mathtt{dl4}}$	$\mathrm{p}_{\mathtt{dl2}}$
		d	$\mathrm{d}_{\mathtt{mi5}}$	$\mathrm{d}_{\mathtt{zsad}}$	$\mathrm{d}_{\mathtt{mi5}}$	$\mathrm{d}_{\mathtt{sad}}$
		MAE / °	5.47 / 28.11	5.47 / 43.73	5.47 / 8.59	5.47 / 9.37

contrast preprocessing $\mathrm{p}_{\mathtt{dl4}}$ with a patch size of $w' = 15$ in combination with a large number of bins is preferable. Although the compass is less accurate under changes of the illumination, we still expect it to be accurate enough to allow for reliable yet not perfect robot navigation. We consider mutual information to be not applicable on a real cleaning robot.

`pp-other` For *loop-closure detection*, different methods perform best depending on the performance measure. The combination of the dividing by variance preprocessing function $\mathrm{p}_{\mathtt{dv4/4}}$ and the cross-correlation $\mathrm{d}_{\mathtt{cc}}$ achieves the best AUC values (1.000 for `day|day` and 0.987 for `day|night`). With respect to robustness against perceptual aliasing and perceptual variability, the combination of first-order edge detectors ($\mathrm{p}_{\mathtt{pw}}$ and $\mathrm{p}_{\mathtt{sob}}$) with the scaled sum of absolute differences $\mathrm{d}_{\mathtt{ssad}}$) performs best. It achieves a perfect loop-closure detection $P_{\mathtt{corr}}$ for the `day|day` and the `day|night` test groups. This result is surprising, because we would expect a combination with $P_{\mathtt{corr}} = 100\,\%$ to also achieve an AUC value of 1.0. We currently cannot explain this result, but, after inspecting the used program code for several times, we consider a programming error to be unlikely. One possible explanation are differences how AUC and $P_{\mathtt{corr}}$ are computed: for the AUC, the dissimilarity values $\hat{\ell}_{i,j}$ are pooled over all databases of a test group, whereas for computing $P_{\mathtt{corr}}$, $P_{\mathtt{misc}}$, and $P_{\mathtt{misf}}$ we first count the occurrences for each of the three cases on a per-database level and then normalize by the number of image pairs contained in the test group.

Regarding *compass accuracy*, the best results were obtained for preprocessing by the local contrast with a patch size of $w' = 7$ and for comparing images by the sum of absolute differences $\mathrm{d}_{\mathtt{sad}}$. The accuracy is negatively influenced if changes of the illumination occur, but with a MAE of approximately 9° we still consider it to be applicable for most real-robot tasks. The methods of this class are computationally efficient alternatives to the methods of the `pp-mi` group (both for loop-closure detection and as a visual compass).

Based on these results, we conclude that accurate and robust loop-closure detection for various workspaces as covered by our image databases is possible with the proposed methods. We therefore did not evaluate the performance measures on a per-database level, but only pooled over databases with similar properties. For small or moderate changes of the illumination as contained in the `day|day` test group, perfect results are obtained for comparing images without prior preprocessing. Although not necessary for achieving perfect results, image preprocessing does not negatively influence the performance. In case strong changes of the illumination are likely to occur during the robot's task (`day|night` test group), image preprocessing prior to image comparison can drastically improve the performance of the proposed methods. With computing times of approximately 10 ms, the effort of applying image preprocessing functions is relatively small. Comparing images by mutual information $\mathrm{d}_{\mathtt{mi}}$ achieves the best results in all considered cases and evaluations. With more than 600 ms, the time required to compute a single image comparison by mutual information is approximately 25 times larger than for other dissimilarity functions relying e.g. on pixel differences. As some of the combinations relying on such dissimilarity functions obtain identical or only slightly worse results, the mutual information has to be considered a baseline method, and these combinations should be preferred for applications with limited computational power.

5.5. Results of Accelerated Compass Method

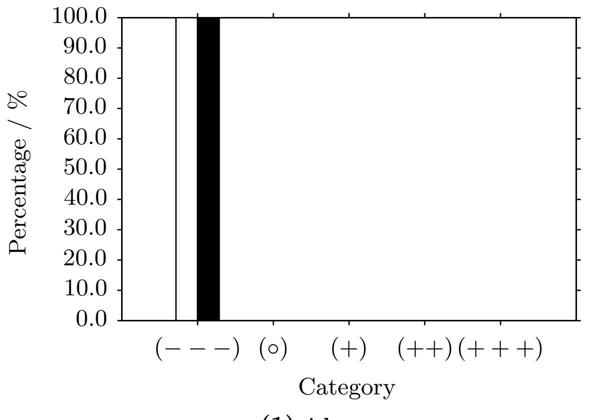
In this section, we describe the results obtained for the accelerated compass method proposed in section 5.2.3; the experimental procedure and the evaluation were performed according to section 5.3. Sections 5.5.1 to 5.5.3 describe the results of evaluating the method's loop-closure detection performance, its compass accuracy and the computational effort of comparing images. Section 5.5.4 summarizes and discusses the results described in the following.

With the parameter combinations as explained in section 5.3.1.2, we test a total of 2072 different methods (i.e. different combinations of preprocessing functions and different parameterizations of the compass) with two *test groups* and evaluate the results with several *evaluation methods*. The test group `day|day` contains approximately 500,000 image pairs taken from image databased collected during the day under (nearly) constant illumination conditions, whereas `day|night` contains approximately 108,000 image pairs taken from cross databases simulating strong illumination changes (section 5.3.1.1). Following the evaluation of the standard compass methods (section 5.4), we reduce the amount of data presented here by splitting the methods into two different *evaluation* groups. The groups were chosen depending on whether images are compared by the compass method with or without prior preprocessing. We refer to these evaluation groups as `id` (without preprocessing) and `pp` (with preprocessing). For each of the two evaluation groups, we identify the combination of image preprocessing and compass parameters which achieves a best possible performance for *both* test groups.

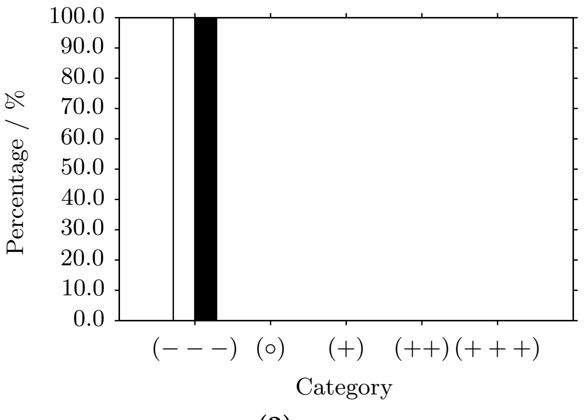
5.5.1. Performance of Loop-Closure Detection

For evaluating the loop-closure detection performance of the accelerated compass method, we follow the approach described in section 5.3.2.1. To get a first impression of the data, we grouped the AUC values into five categories as defined in table 5.3. Figure 5.20 visualizes the resulting distribution of AUC values for each of the two evaluation groups. In all cases AUC values below 0.6 (– – –) are obtained making loop-closure detection impossible. With these results, it is not reasonable to identify the best methods of each evaluation group and to further analyze the AUC values.

In a second evaluation step, we visualize for each database the range of dissimilarity values for positive and negative samples, i.e. for image pairs taken at identical positions in space and pairs acquired at different positions. By this means, we analyze whether or not loop-closure detection is possible, i.e. if if positive and negative samples are separable, on the level of single databases.

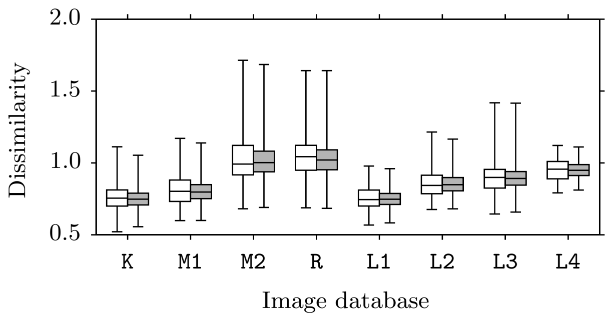


(1) `id`

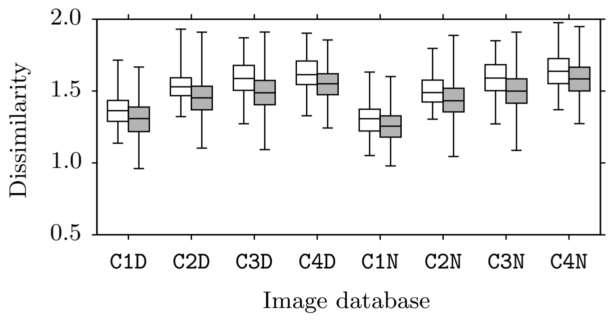


(2) `pp`

Figure 5.20.: Frequency distributions of AUC values for the accelerated compass variant. The distributions are used to identify promising methods for loop-closure detection as described in section 5.3.2.1. Subfigures (1) and (2) show the frequency distributions computed for the two *evaluation* groups `id` and `pp`. White and black bars depict the distributions for the *test* groups `day|day` and `day|night`, respectively. Binning was done w.r.t. the categories defined in table 5.3 as follows: (− − −) loop-closure detection not possible; (∘) moderate loop-closure detection performance; (+) good performance; (++) very good performance; (+ + +) ideal results (i.e. AUC = 1.0).



(1) Best combination ($p_{\texttt{dl4}}$, $r = 1$, $c = 4$) of `day|day` group with AUC = 0.580



(2) Best combination ($p_{\texttt{dv4/4}}$, $r = 1$, $c = 8$, AUC = 0.574) of `day|night` group with AUC=0.574. Dissimilarity values are scaled with a factor of 10^3.

Figure 5.21.: Database-wise analysis of accelerated compass variant w.r.t. loop-closure detection performance. For each database, we identified the combination of preprocessing and compass parameters with the largest AUC value (table C.3), and analyzed the resulting dissimilarity values $\hat{\ell}_{i,j}$. Subfigures (1) and (2) visualize the results for the databases of the `day|day` and `day|night` group, respectively. Positive samples (loop closures) are depicted by white boxes; negative samples (different positions of image acquisition) are depicted by gray boxes. The boxes cover 50 % of the samples ranging from the lower to the upper quartile. The median is marked by the horizontal black line. The whiskers span the range from the smallest to the largest dissimilarity value. The figures clearly reveal that positive and negative samples cannot be distinguished therefore making loop-closure detection impossible. Database names are abbreviated according to table 5.2.

Figure 5.21 shows the results for analyzing the combination of preprocessing function p, number of rings r, and number of Fourier coefficients c which obtained for each database the largest AUC values among all tested combinations. In all cases, positive and negative samples cannot be separated because overlapping dissimilarity values were obtained for positive and negative samples. Furthermore, the ranges of dissimilarity values vary strongly depending on the used databases. For these reasons, the proposed method cannot detect loop closures reliably. We therefore do not evaluate the method's robustness against perceptual aliasing and perceptual variability. Possible solutions to this problem are left for future work and outlined in section 5.6.

Table 5.11.: Best combinations of accelerated compass method w.r.t. compass accuracy. Subtables (1) and (2) contain for the two *evaluation* groups `id` and `pp`the median angular error (MAE) and the average angular error (AAE) computed for the `day|day` and `day|night` *test* groups. The highlighted columns mark the measure used for sorting. In case several combinations achieve identical MAE values, the AAE was used as secondary sorting criterion. for the evaluation group `pp` (subtable (2)), the optima for the `day|day` and the `day|night` test groups are obtained by different combinations. The upper part of subtable (2) thus contains the best method for the `day|day` group, the lower part contains the best method for the `day|night` group.

(1) `id`

				Test groups			
				`day|day`		`day|night`	
p	r	b	$r \cdot b$	MAE / °	AAE / °	MAE / °	AAE / °
$\mathrm{p}_{\texttt{id}}$	64	8	512	5.47	14.44	49.20	69.45
$\mathrm{p}_{\texttt{id}}$	32	8	256	5.47	14.56	49.20	69.45
$\mathrm{p}_{\texttt{id}}$	64	230	14 720	5.47	14.69	51.15	69.94
$\mathrm{p}_{\texttt{id}}$	64	128	8192	5.47	14.71	51.15	69.96
$\mathrm{p}_{\texttt{id}}$	64	96	6144	5.47	14.71	51.15	69.95
$\mathrm{p}_{\texttt{id}}$	64	16	1024	5.47	14.72	51.15	70.17
$\mathrm{p}_{\texttt{id}}$	64	32	2048	5.47	14.72	51.15	69.83
$\mathrm{p}_{\texttt{id}}$	64	64	4096	5.47	14.73	51.15	69.90
$\mathrm{p}_{\texttt{id}}$	32	16	512	5.47	14.82	51.15	70.08
$\mathrm{p}_{\texttt{id}}$	32	32	1024	5.47	14.85	51.15	69.78
$\mathrm{p}_{\texttt{id}}$	16	8	128	5.47	14.91	49.98	70.13
$\mathrm{p}_{\texttt{id}}$	16	16	256	5.47	15.11	51.93	70.76
$\mathrm{p}_{\texttt{id}}$	8	8	64	5.47	15.93	49.59	70.34
$\mathrm{p}_{\texttt{id}}$	8	16	128	5.47	16.06	51.54	70.84

(2) `pp`

				Test groups			
				`day|day`		`day|night`	
p	r	b	$r \cdot b$	MAE / °	AAE / °	MAE / °	AAE / °
$\mathrm{p}_{\texttt{pw}}$	8	8	64	3.91	17.91	66.77	77.03
$\mathrm{p}_{\texttt{dl2}}$ $(w' = 7)$	1	8	8	5.47	20.00	12.50	34.02
$\mathrm{p}_{\texttt{dl3}}$ $(w' = 11)$	1	8	8	5.47	18.16	12.50	36.76
$\mathrm{p}_{\texttt{dl3}}$ $(w' = 15)$	1	4	4	8.59	18.43	12.50	26.76

5.5.2. Compass Accuracy

In order to assess the compass accuracy of the proposed method, the median and mean angular errors MAE and AAE were computed as described in section 5.3.2.3. For each of the two *evaluation* groups (`id` and `pp`) we search for a combination of preprocessing method and compass parameterization which achieves best possible results for *both* test groups `day|day` and `day|night`. The best combinations for each evaluation group are summarized in table 5.11, and figures 5.22 to 5.24 show a more detailed evaluation of the compass accuracy depending on the dimensionality of the compass method.

For the `id` group comparing images without prior preprocessing (table 5.11.1), the best combination achieves a MAE of 5.47° and an AAE of 14.44° for the `day|day` group. The results for the `day|night` group are considerably worse with the MAE and AAE being approximately 50° and 70°. These values were obtained for $r = 64$ rings and $b = 8$ Fourier coefficients resulting in a dimensionality of $r \cdot b = 512$. Figure 5.22 visualizes the MAE depending on the dimensionality $r \cdot b$ with different choices of r being coded by different markers. For the `day|day` group (figure 5.22.1), the optimal MAE of 5.47° is achieved for a dimensionality of 64 or more dimensions. Combinations with a

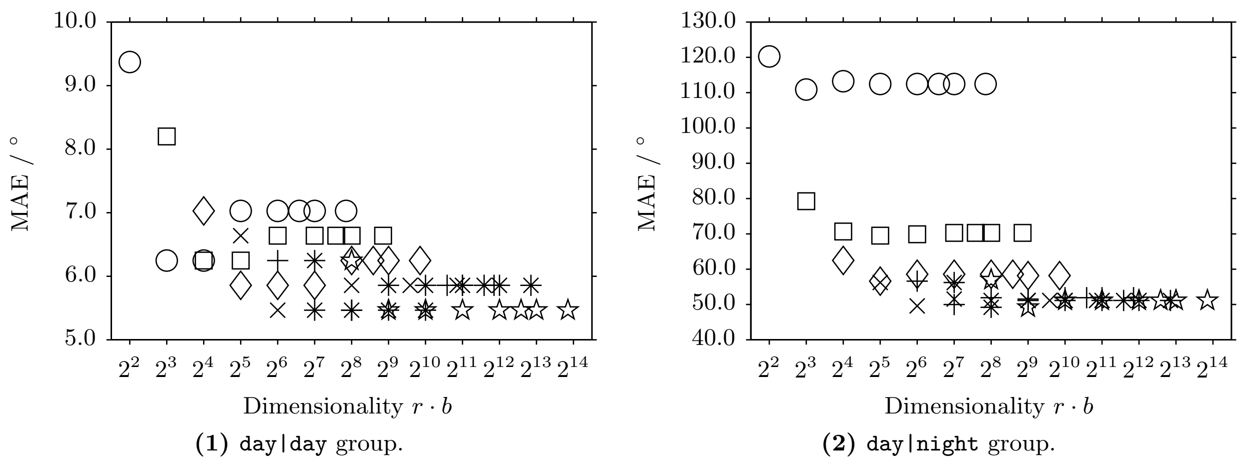


Figure 5.22.: Compass accuracy of accelerated compass variant depending on its dimensionality for comparing images without prior preprocessing (*evaluation* group `id`). Subfigures (1) and (2) show the results for the `day|day` and the `day|night` *test* groups. As an accuracy measure, the median angular error (MAE) is visualized depending on the dimensionality $r \cdot b$ of the compass method. Images are compared without prior preprocessing (evaluation group `id`). The number of rings r is coded by different markers as follows: (○) $r = 1$, (□) $r = 2$, (◇) $r = 4$, (×) $r = 8$, (+) $r = 16$, (∗) $r = 32$, (☆) $r = 64$. Please note the different scales of the vertical axes.

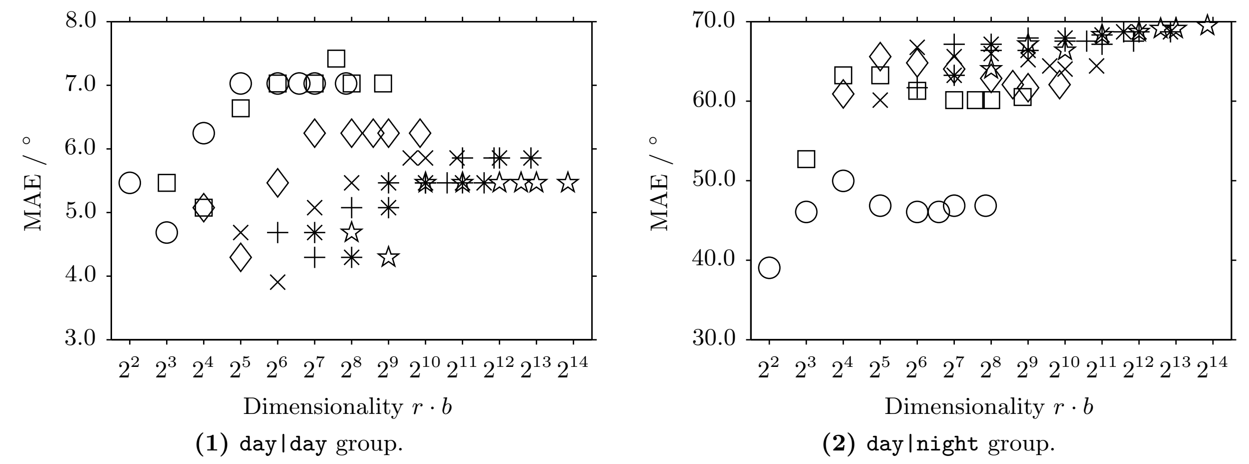


Figure 5.23.: Compass accuracy of accelerated compass variant depending on its dimensionality for images preprocessed by Prewitt filter p_{pw}. The subfigures (1) and (2) visualize the results for the two test groups `day|day` and `day|night`, respectively. As an accuracy measure, the median angular error (MAE) is visualized depending on the dimensionality $r \cdot b$ of the compass method. The number of rings r is coded by different markers as follows: (○) $r = 1$, (□) $r = 2$, (◇) $r = 4$, (×) $r = 8$, (+) $r = 16$, (∗) $r = 32$, (☆) $r = 64$. Please note the different scales of the vertical axes.

small number of rings ($r = 1$ (○) and $r = 2$ (□)) obtain larger errors than methods with a larger number of rings. Despite the much larger angular errors, the same holds for the `day|night` group (figure 5.22.2).

For comparing preprocessed images (`pp`; table 5.11.2), the optimal combinations of the preprocessing function p, the number of rings r, and the number of Fourier coefficients b differ for the `day|day` and the `day|night` group. With an MAE of 3.91°, the combination of preprocessing by the Prewitt filter p_{pw} with $r = 8$ and $b = 8$ achieves the optimal performance for the `day|day` group; the corresponding AAE is 17.91°. For the `day|night` group, the error of this combination increases to a MAE and an AAE of approximately 67° and 77°, respectively. Figure 5.23 shows the influence of the dimensionality on the compass accuracy for preprocessing by the Prewitt filter p_{pw}. Regarding

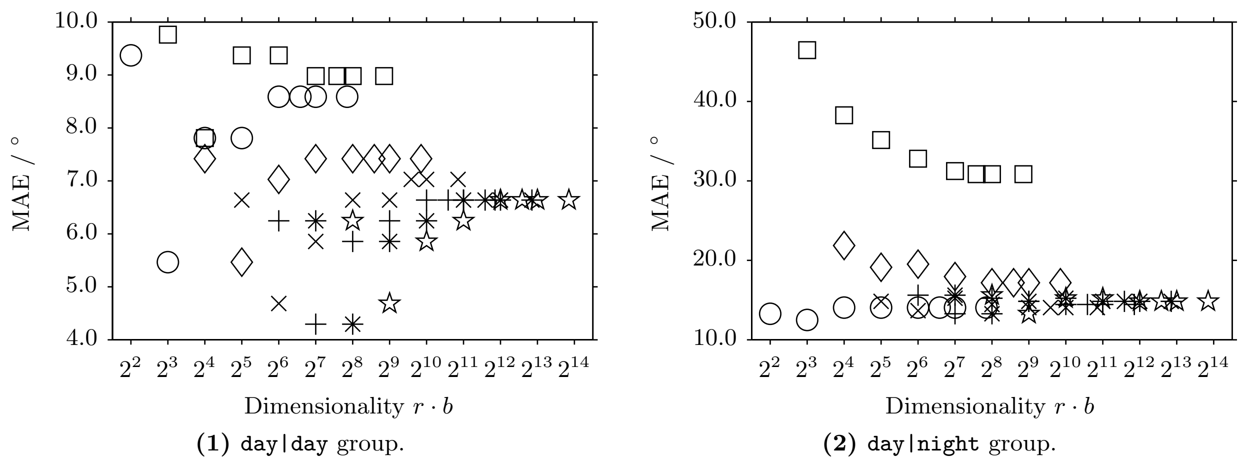


Figure 5.24.: Compass accuracy of accelerated compass variant depending on its dimensionality for comparing images preprocessed by local contrast $p_{\mathtt{dl2}}$ with window size $w' = 7$. The subfigures (1) and (2) visualize the results for the two test groups `day|day` and `day|night`, respectively. As an accuracy measure, the median angular error (MAE) is visualized depending on the dimensionality $r \cdot b$ of the compass method. The number of rings r is coded by different markers as follows: (○) $r = 1$, (□) $r = 2$, (◇) $r = 4$, (×) $r = 8$, (+) $r = 16$, (∗) $r = 32$, (☆) $r = 64$. Please note the different scales of the vertical axes.

the `day|day` group (figure 5.23.1), the optimum MAE of 3.91° is obtained for a single combination, but several other combinations achieve only slightly worse results. It seems that the best results are achieved by combining a moderate number of rings and a moderate number of Fourier coefficients. In most of the cases, combinations with a small number of rings ($r \in \{1, 2, 4\}$, (○), (□), and (◇)) achieve relatively large angular errors. Combinations with a dimensionality larger than 2^{10} only achieve a mediocre accuracy. Analyzing the influence of the dimensionality $r \cdot b$ onto the MAE for the `day|night` group reveals completely different results (figure 5.23.2). In this case, the best combinations with MAEs ranging from approximately 40° to 50° are obtained for using only a single ring (○). For more than 2 rings and an overall dimensionality ranging from 2^4 to 2^{10}, all combinations achieve angular errors between 60° and 70°; further increasing the dimensionality results in MAE values being close to 70°.

If the optimal combination is chosen depending on the MAE obtained for the `day|night` group, three combinations achieve an optimal MAE of 12.20° and an AAEs between 26.76° and 36.76° (table 5.11.2). The combinations all rely on the dividing by luminance preprocessing $p_{\mathtt{dl}}$ with a patch size w' of 7 or 11 pixels and have dimensionalities between 4 and 8. For the `day|day` group, these combinations all achieve a MAE of 5.47° and AAEs ranging from 9.41° to 14.44°. Figure 5.24 visualizes how the dimensionality influences the resulting compass accuracy for preprocessing by $p_{\mathtt{dl2}}$. For the `day|night` group (figure 5.24.2), the best MAE value of 12.50° is only obtained for $r = 1$ (○) and $b = 8$. The combinations relying on $r = 2$ (□) and on $r = 4$ (◇) perform considerably worse; especially for using $r = 2$ rings, the MAEs are above 30°. For a larger number of rings, a good but not optimal accuracy is obtained with MAE values of approximately 15°. Regarding the `day|day` group (figure 5.24.1), the best accuracy is achieved for a dimensionality of 2^7 and 2^8 in conjunction with $r = 16$ (+) and $r = 32$, (∗), respectively. Like for other considered cases, combinations relying on a smaller number of rings achieve in most cases worse MAE values, and increasing the dimensionality leads to a moderate accuracy of approximately 6.5°.

On the basis of these results, we can conclude that the accelerated compass variant proposed in this chapter is capable of estimating the change of orientation between two panoramic images. For small or moderate changes of the illumination like those occurring in the `day|day` group, the method is accurate; if stronger changes occur —as it is the case for the `day|night` group— the

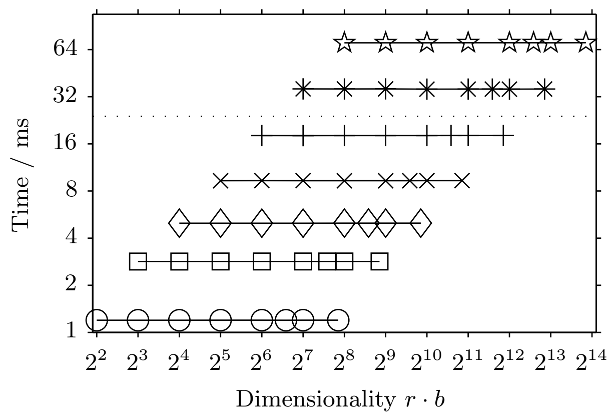


Figure 5.25.: Computing times of accelerated compass method. The plot shows the computing time against the overall number of coefficients $r \cdot b$ with r being the number of subimages and b being the number of Fourier coefficients. The dotted line marks the computing time required for a pixel-by-pixel comparison with the cross-correlation function d_{cc}. The number of rings r is coded by different markers as follows: (∘) $r = 1$, (□) $r = 2$, (⋄) $r = 4$, (×) $r = 8$, (+) $r = 16$, (∗) $r = 32$, (☆) $r = 64$. Please not the logarithmic scale of the vertical axis.

larger errors occur. This effect is only partially compensated by image preprocessing.

5.5.3. Computational Aspects

As the accelerated compass variant used the same image preprocessing methods as the standard method, we only measure the time required to compare two preprocessed images. The results of this evaluation are shown in figure 5.25 and table C.2. In the figure, the computing time is plotted against $r \cdot b$, i.e. the number of subpanoramas r times the number of Fourier coefficients b. The number of rings r is coded by different markers as explained in figure 5.25. For fixed r, the number of coefficients b increases from left to right according to equation (5.64). For more than 2 rings, the required computing time depends linearly on the number of rings. As the used function for the discrete Fourier transformation computes all Fourier coefficients and not only the b required coefficients (section 5.3.1.3), the computing time solely depends on the number of rings r (section 5.3.1.3). Thus, the resulting computing times are 1.2 ms for $r = 1$ (∘), 2.8 ms for $r = 2$ (□), 4.9 ms for $r = 4$ (⋄), 9.3 ms for $r = 8$ (×), 18.1 ms for $r = 16$ (+), 35.8 ms for $r = 32$ (∗), and 70.7 ms for $r = 64$ (☆).

To this end, the time required to compute the visual compass and the image dissimilarity spans a large range. For applications of the method, this allows to flexibly adjust the number of rings r depending on the robot's task, the available computational power, and the available computing time. In comparison to the standard cross-correlation function d_{cc}, which requires approximately 24 ms (dotted line in figure 5.25), the accelerated method is faster for up to $r = 16$ rings. For a larger number of rings, the accelerated variant performs worse. By using a variant of the FFT computing only the b required coefficients, we expect that combinations with more rings but less coefficients can be found which can be computed in less than 24 ms.

5.5.4. Discussion and Conclusions

Based on the results described in sections 5.5.1 to 5.5.3, we can conclude that the proposed method is applicable as visual compass but not for loop-closure detection. For the latter application, the dissimilarity values computed for positive samples (images taken at identical positions in space) and for negative samples (images acquired at different positions in space) have to be separable. This is not the case for the proposed method (figure 5.21). We expect that this is due to summing the residuals of the maxima obtained for each ring in order to determine the overall image dissimilarity (equation (5.51)). In contrast to the accelerated method, the dissimilarity value computed by the standard method is the residual of an optimization searching for a single optimum for the entire image. In this case the resulting image dissimilarity is a trade-off which is influenced by good-matching image regions and ambiguous or mismatching regions. We currently think that this way of optimization is better suited for loop-closure detection (section 5.7.2).

Regarding its application as a visual compass, the method achieves good results, but its performance decreases considerably under strong changes of the illumination. Since the accelerated method relies on the cross-correlation function $\mathrm{d}_{\mathtt{cc}}$ for computing image dissimilarities (section 5.2.3), it cannot be expected that the accelerated variant is more tolerant against illumination changes than the original method. If strong changes of the illumination are unlikely during the robot's task, preprocessing images with the Prewitt filter $\mathrm{p}_{\mathtt{pw}}$ yields a better compass accuracy than comparing unprocessed images. However, for strong changes of the illumination, this combination performs worse than comparing images without prior preprocessing. In case strong changes are expected to occur, preprocessing images through the dividing by luminance preprocessing function $\mathrm{p}_{\mathtt{dl}}$ can considerably improve the compass accuracy while only slightly decreasing the accuracy under nearly constant illumination conditions.

With respect to computational complexity, we can conclude that the computing time required to compare two images linearly depends on the number of rings r. For up to 16 rings, two images can be compared in less than 20 ms. The effort of the accelerated compass variant exceeds that of the standard method (except for comparing images by the mutual information) if more than 16 rings are used. By applying a FFT algorithm only computing the required number of Fourier coefficients, this drawback could be circumvented.

5.6. Overall Discussion and Conclusions

In this chapter, we approached the loop-closure problem by a combination of image preprocessing techniques to reduce the influence of changes of the illumination and of global image comparisons to classify whether or not two images are identical. Since global image dissimilarity functions require the images to be aligned w.r.t. a common reference direction, we integrated the loop-closure detection into the visual compass proposed by Zeil, Hoffmann, and Chahl [718]. We also proposed a variant of the visual compass referred to as "accelerated compass variant" (section 5.2.3) which approximates the computation of the cross-correlation function in the Fourier domain. Computing the correlation in the Fourier domain allows for computing the compass shift and the image dissimilarity without repeatedly shifting one of the images as is the case for the standard compass method operating in the image domain. The proposed method approximates the cross correlation $\mathrm{d}_{\mathtt{cc}}$ because (i) it uses only the b largest Fourier coefficients and (ii) computes the dissimilarity from a set of r 1D subpanoramas. These two aspects in combination with computing the correlation in the Fourier domain constitute the efficiency of the method.

The remainder of this section is structured as follows: in section 5.6.1, the standard compass method and the accelerated compass variant proposed in this chapter are compared, and in section 5.6.2, conclusions about the applicability of the proposed methods on a real cleaning robot are discussed.

5.6.1. Comparison Between Standard and Accelerated Compass Methods

This section compares the standard compass method (section 5.2) and the newly proposed accelerated compass variant (section 5.2.3) with respect to loop-closure detection performance and compass accuracy. Currently, only the standard compass method is suitable for loop-closure detection. As the accelerated variant sums the residual remaining for each ring in order to compute the overall image dissimilarity, the dissimilarities obtained for positive and negative class samples cannot be separated (figure 5.21 and section 5.4.6). An improvement to circumvent this drawback is proposed in section 5.7.

Table 5.12 summarizes the results obtained for analyzing the compass accuracy of the standard compass method and the accelerated compass variant. For comparing unprocessed and preprocessed

Table 5.12.: Comparison between standard compass method and accelerated compass method w.r.t. compass accuracy. Each table includes the results of three methods which are (i) the best standard compass method with arbitrary image comparison, (ii) the best standard compass method comparing images with the cross correlation $d_{\mathtt{cc}}$, and (iii) the best accelerated compass method (i.e. comparing images by $d_{\mathtt{fcc}}$). The subtables (1) and (2) contain the methods for comparing unprocessed and preprocessed images (*evaluation* groups `id` and `pp`). For the evaluation group `pp`, different optima result for the `day|day` and the `day|night` *test* groups. The upper part of subtable (2) thus contains the best method for the `day|day` group, the lower part contains the best method for the `day|night` group. In case several combinations achieve identical MAE values, the corresponding AAE is used for sorting.

(1) `id`

		Test groups			
		`day|day`		`day|night`	
p	d	MAE / °	AAE / °	MAE / °	AAE / °
$p_{\mathtt{id}}$	$d_{\mathtt{mi5}}$ ($b = 64$)	5.47	9.41	28.11	42.77
$p_{\mathtt{id}}$	$d_{\mathtt{fcc}}$ ($r = 8$, $b = 8$)	5.47	14.44	49.20	69.45
$p_{\mathtt{id}}$	$d_{\mathtt{cc}}$	6.25	13.29	60.91	78.88

(2) `pp`

		Test groups			
		`day|day`		`day|night`	
p	d	MAE / °	AAE / °	MAE / °	AAE / °
$p_{\mathtt{pw}}$	$d_{\mathtt{fcc}}$ ($r = 8$, $b = 8$)	3.91	17.91	66.77	77.03
$p_{\mathtt{dog3}}$ ($\sigma_1 = 0.01$, $\sigma_2 = 0.08$)	$d_{\mathtt{mi5}}$ ($b = 64$)	4.69	9.70	10.15	23.65
$p_{\mathtt{heq5}}$ ($b = 64$)	$d_{\mathtt{cc}}$	5.47	11.76	43.73	53.20
$p_{\mathtt{dl4}}$ ($w' = 15$)	$d_{\mathtt{mi5}}$ ($b = 64$)	5.47	9.17	8.59	16.13
$p_{\mathtt{dlc4}}$ ($w' = 15$)	$d_{\mathtt{cc}}$	7.03	13.82	9.37	23.93
$p_{\mathtt{dl3}}$ ($w' = 11$)	$d_{\mathtt{fcc}}$ ($r = 1$, $b = 8$)	5.47	20.00	12.50	34.02

images (tables 5.12.1 and 5.12.2), each subtable contains the best results obtained for the standard method comparing images with any of the tested dissimilarity functions, for the standard method comparing images with the cross-correlation $d_{\mathtt{cc}}$, and for the accelerated compass variant approximating $d_{\mathtt{cc}}$ by $d_{\mathtt{fcc}}$. This allows us to compare the proposed compass method with the two parameterizations of the standard compass method which are either most accurate or most similar to the proposed method.

For comparing images without prior preprocessing (table 5.12.1), the accelerated compass method achieves for the `day|day` *test* group the same median angular error (MAE) of 5.47° than the best standard method. Both methods perform slightly better than the best standard method comparing images with the cross-correlation function $d_{\mathtt{cc}}$. Regarding the `day|night` test group, all methods perform considerably worse and achieve MAEs of 28° and more. These large errors make them not applicable for robot navigation tasks. For comparing preprocessed images (table 5.12.2), different methods yield the best results depending on whether the MAEs of the `day|day` group or that of the `day|night` group are used for ranking. In the former case, the proposed compass method performs achieves for the `day|day` group a better MAE than the other two methods. However, the method (i.e. preprocessing by Prewitt filter $p_{\mathtt{pw}}$ and a compass with $r = 8$ subpanoramas and $b = 8$ Fourier coefficients) performing best for the `day|day` group performs worst for the `day|night` group. With 66°, the MAE of the accelerated method is much worse than that of the standard methods relying on the cross correlation $d_{\mathtt{cc}}$ (MAE ≈ 44°) and that of the best standard method (MAE ≈ 10°). If the methods are ranked depending on the MAE values of the `day|night` group, the best methods achieve a good performance for both test groups. For the `day|day` group, the best standard method and the accelerated method both achieve a MAE of 5.47°; the best standard method comparing images by cross correlation $\mathrm{d}_{\mathtt{cc}}$ achieves an MAE of approximately 7°. Regarding the median angular errors for the `day|night` group, the two standard methods perform better with MAEs of approximately 9°. The accelerated compass method performs worst for this test group and achieves a MAE of 12.5°.

Based on this comparison, we conclude that the newly proposed compass variant is competitive for small or moderate changes of the illumination as contained in the `day|day` data set. For this case, it achieves the same or even slightly better results than the standard method comparing images by mutual information; However, if strong changes of the illumination occur during the robot's task, the accelerated compass variant achieves larger angular errors than the best combinations of the standard method. In all considered cases, the accelerated compass method is more efficient than the standard method relying on mutual information as dissimilarity function. If other dissimilarity functions are used, the accelerated compass variant is with its current implementation more efficient for $r = 16$ or less subpanoramas.

5.6.2. Applicability on a Real Cleaning Robot

Based on the measurements of the required computing time (sections 5.4.4, 5.4.5 and 5.5.3), theoretical conclusions can be drawn if the methods proposed in this chapter can be applied on a real cleaning robot. We assume that the robot moves with 10 cm/s and that it takes a new snapshot every 10 cm. These conditions are comparable to the experimental conditions under which the real-robot experiments described in (section 6.4 and chapter 4) were conducted. Since loop-closures have to be detected in the time between adding to consecutive snapshots to the map, there is at most 1 s of computing time available for this purpose. For the further evaluation, we do not take into account (i) that the cleaning robot has to concurrently perform further processing steps such as visual homing or updating its position estimates and (ii) that the robot's on-board computer probably has less computational power than the desktop computer used for the experiments described in this chapter.

With a computing time of 600 ms for a single image comparison, mutual information is not applicable on a real robot and has to be considered a baseline method. We further assume that image preprocessing requires 10 ms and that an image comparison requires approximately 25 ms (sections 5.4.4 and 5.4.5). Thus, 39 images can be compared if the preprocessed images are stored in the topological map; otherwise, only 28 images can be compared due to repeated applications of the image preprocessing functions. 39 images correspond to a single cleaning lane with a length of 3.8 m. If one analyzes footprints of real apartments, as e.g. depicted in figures 7.3 and B.4, one clearly sees that cleaning lanes of approximately 4 m length can occur even in mid-sized apartments. For safe loop-closure detection in larger environments, we think that —including some safety tolerance— 80 to 100 comparisons are necessary. Thus, at least with the current implementation, the proposed methods are not applicable on a real cleaning robot; improvements making the methods applicable on a real cleaning robot therefore have to halve the current computing time and are discussed in section 5.7.1.

5.7. Future Working Directions

Future working directions related to holistic loop-closure detection methods include (i) optimizing the current rapid-prototyping implementation (section 5.7.1), (ii) to improve the Fourier-based compass method (section 5.7.4), (iii) to test further image preprocessing and image comparison functions (section 5.7.4), and (iv) to apply the concept of rings also to the standard method and to increase the method's accuracy by turning it into a probabilistic (section 5.7.3). As loop-closure detection based on global image dissimilarities is closely related to signature-based loop-closure detection (chapter 6), several of the future working directions described in section 6.6 can also be pursued for the methods proposed in this chapter. These include topological mapping and inferring spatial information based on pairwise image dissimilarities or reintroducing metrical position information based on pairwise image dissimilarities. However, we feel that signature-based approaches are better suited for these approaches because these methods allow for more efficient image comparisons.

5.7.1. Efficient Implementation

The aim of this working direction is to make the holistic loop-closure detection methods applicable on a real cleaning robot. A first step is to reimplement the proposed preprocessing and image comparison functions in integer arithmetic. A second step is to speed up the computation of image-dissimilarity values by relying on precomputed look-up tables (e.g. for computing pixel differences) or early termination of summations. Furthermore, as both image preprocessing and computing image dissimilarities involve many operations which can be executed in parallel, further optimizations could include to reimplement the methods based on Streaming SIMD Extensions (SSE; e.g. [I50]). With this programming technique, Prof. Dr. Ralf Möller could recently show that different (but nevertheless closely related) image dissimilarity functions can be evaluated in less than 10 ms (Möller [450]).

5.7.2. Improvement of the Accelerated Compass Variant

The primary goal of future work should be to improve the accelerated compass method in order to make it applicable for loop-closure detection. In the current implementation, the overall image dissimilarity $\hat{\ell}$ is computed by summing the residuals $\hat{\ell}_j$ obtained for every subpanorama: $\hat{\ell} = \sum_j \hat{\ell}_j$ (equation (5.51)). As already discussed in section 5.5.4, we expect that the method will be suitable for loop-closure detection if equation (5.51) is replaced by a method making the overall dissimilarity $\hat{\ell}$ dependent on the resulting *overall* compass shift $\hat{s}$ (rather than on that of *each subpanorama*):

$$\hat{\ell} = \sum_{j=0}^{r-1} \ell_j(\hat{s}). \tag{5.73}$$

5.7.3. Subdivision of the Panoramic Image Into Rings

To apply the concept of rings onto the standard compass method (Zeil, Hoffmann, and Chahl [718]), the panoramic image has to be divided into subpanoramas as described in section 5.2.3. For each subpanorama, the compass shift and the compass residual is computed according to equations (5.2) and (5.4). The estimates obtained by this means can then be fused to an overall estimate of the orientation change (equation (5.50)) and an overall dissimilarity measure (equation (5.73)). By this extension, we expect that the compass accuracy of the standard method can be further improved because the overall compass shift is derived from several estimates, which should be more robust against outliers. Furthermore, the extension allows for comparing images by all tested image dissimilarity functions and is not restricted to the approximation of the cross-correlation function as is the case for our accelerated compass variant. However, it requires to repeatedly shift and evaluate the image dissimilarity function.

Turning the currently used methods into probabilistic methods is supposed to increase their compass accuracy and loop-closure detection performance by reducing the influence of outliers and ambiguous matches. Instead of fusing the j $(0 \leq j < r)$ estimates $\hat{s}_j$ of the compass shift obtained for every ring by computing their median (equation (5.50)), the probability density function of a von Mises distribution

$$\boldsymbol{m}_j = \text{mises}(s_j, \sigma_j^2) \tag{5.74}$$

centered at the maximum s_j and with variance σ_j^2 can be computed. The von Mises distribution is also referred to as circular normal distribution (textbook: [32]). In the discrete case considered here, $\boldsymbol{m}_j$ is sampled at discrete positions corresponding to the pixel positions in the ring-shaped image. Fusing the estimates of every ring is then done by multiplying all $\boldsymbol{m}_j$ and finding the maximum of the resulting product:

$$\hat{s} = \max_{s=0}^{w-1} \prod_{j=0}^{r-1} \boldsymbol{m}_j. \tag{5.75}$$

5.7.4. Testing Further Preprocessing and Image-Dissimilarity Functions

The third direction of future work is to test preprocessing methods and dissimilarity functions which have not been considered yet. As an alternative preprocessing method, histogram warping could be applied [123, 259, 260]. For this purpose, the gray-value distribution of one image is warped in order to best possibly fit the gray-value distribution of the other image (rather than equalizing the gray-value distribution of both images as is the case for histogram equalization; section 5.2.1). Image dissimilarities are then computed for the image with the warped gray-value distribution and the reference image. Although we are not aware of an application of this technique for visual robot navigation, we expect it to be an interesting alternative to histogram equalization. In particular if the image histograms are similar, we expect it to achieve better results than histogram equalization emphasizing contrasts for dominating intensity ranges and reducing contrasts for less frequent intensity ranges.

At the current stage of our work, we do not consider local image dissimilarity functions. Although we currently see two drawbacks of such dissimilarity functions, they could be applied for loop-closure detection. First, the computational effort of local image dissimilarity functions is larger because (i) images have to be divided into subregions, (ii) the regions have to be compared, and (iii) the dissimilarity values have to be fused to an overall dissimilarity. The second drawback is that local dissimilarity functions have more internal parameters such as the size, the position, the number, and the shape of subregions. Despite these drawbacks, we expect that local dissimilarity functions can achieve a better robustness against changes of the illumination because several local dissimilarity values are fused to the overall image dissimilarity such that local intensity changes can be tolerated rather than only global ones. For this purpose, the method should be capable of compensating for a certain proportion of outliers or erroneous matches. An alternative to local image dissimilarity functions could be to test the illumination-tolerant correlation measures proposed by Möller [450]. For our 2D warping method (sections 3.5.2.2 and 4.4.2), they exhibit a good robustness against changes of the illumination and can be efficiently computed.

6. Signature-Based Loop-Closure Detection

The methods proposed in this chapter solve the loop-closure problem by deriving and comparing global image signatures instead of comparing entire images as in the previous chapter.
Sections 6.1 and 6.2 introduce and describe the developed methods. The experimental procedure, evaluation methods and results of database and real-robot experiments are described in sections 6.3 and 6.4. The chapter ends with a discussion (section 6.5) and an outlook to future working directions (section 6.6).
This chapter is an extension of the bachelor's thesis of Oliver Schlüter [575] supervised by Lorenz Hillen and Martin Krzykawski. Except for the program to compute AUC values (section 6.3.1.3), which was provided by Oliver Schlüter, the software required to perform the database experiments was implemented by Lorenz Hillen. During his time as student assistant, Oliver Schlüter also implemented the software framework for real-robot experiments under the supervision of Lorenz Hillen. The software for data evaluation and visualization of the real-robot experiments was implemented by Lorenz Hillen who also conducted the experiments. The active visual tracking system used to track the robot during real-robot experiments (figure 6.15) was initially developed by Daniel Venjakob in a student project [668] supervised by Lorenz Hillen. The system was later on improved by Martin Krzykawski (low-level aspects of client-server communication) and Lorenz Hillen (high-level aspects including calibration, robustness against illumination changes, usability, visualization, and data logging).
*Preliminary results of this chapter have already been published as conference paper (*Gerstmayr-Hillen *et al. [223]) and presented as poster (*Gerstmayr-Hillen *et al. [221]).*

6.1. Introduction

Signature-based solutions to the loop-closure problem characterize places by a low-dimensional image-signature derived from the entire image (section 3.3.1.1). To better distinguish global image-descriptors from local feature-descriptors computed at image regions around points of interest (reviews: [226, 372, 438, 439, 577, 644], textbook: [586]), we refer to global image descriptors as image signatures. For detecting loop closures, signature-based methods compare the low-dimensional image signatures and therefore allow considerable more efficient image comparisons than holistic and feature-based loop-closure detection methods. Compared to holistic loop-closure detection methods, signature-based methods are more efficient because (i) the compass step required by holistic methods can be avoided by applying a signature function which is invariant under rotations of the robot, and because (ii) low-dimensional image signatures are compared instead of entire images. Due to the latter aspect, signature-based methods are also more efficient than feature-based methods, which have to be considered as the standard approach to loop-closure detection and explicitly solve the correspondence problem by descriptor matching. For this purpose, several local feature descriptors need to be matched (section 3.3.2.1), each of which has a dimensionality comparable to or even larger than the dimensionality of the signatures proposed in this chapter.

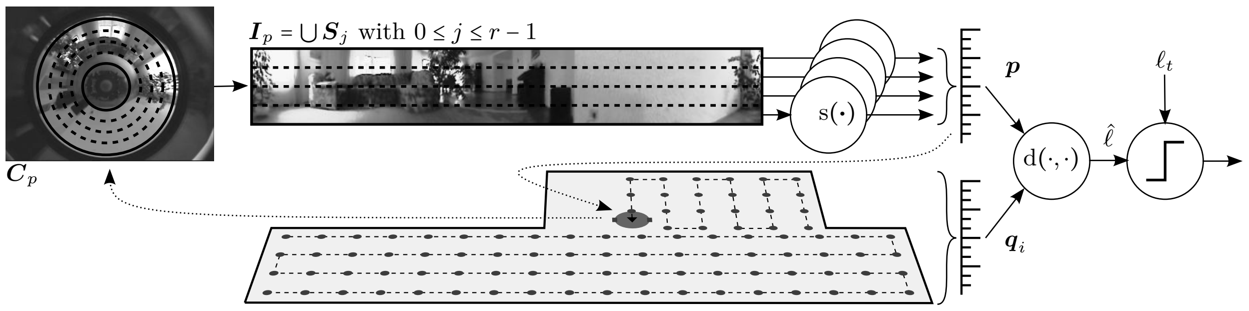


Figure 6.1.: Principles of signature-based loop-closure detection. By covering the robot's workspace with segments of parallel lanes (dashed lines), a dense purely topological map (section 3.6.4.1) is built (edges not shown). Loop closures can occur at the borders of such segments. By applying a signature function s, image signatures $\boldsymbol{p}$ are computed which can be stored in the topological map and can be reused in later processing steps. For loop-closure detection, the current signature $\boldsymbol{p}$ is compared with several image signatures $\boldsymbol{q}_i$ stored in the map by applying a dissimilarity function d. Each comparison results in a dissimilarity value $\hat{\ell}$ which is used for classification w.r.t. a classification threshold ℓ_t. Since the used signature functions are rotationally independent, signatures can be compared independent of the robot's orientation. Thus, signature-based approaches do not require the visual compass step inherent to holistic loop-closure detection methods (chapter 5).

In the context of topological navigation, image signatures such as color histograms (e.g. [476, 647, 688]), gray-value histograms (e.g. [242]), color statistics (e.g. [476]), rotation-dependent eigenspace representations (e.g. [210, 311]), rotation-invariant eigenspace representations (e.g. [313]), Haar integrals (e.g. [94, 352]), and absolute Fourier coefficients (e.g. [169, 171, 172, 421, 427, 506, 530]) have been used (for details on these methods please refer to sections 3.3.1.1, 3.6.3.2 and 3.6.4.1). All these methods have in common that invariance against rotations of the robot is usually achieved by deriving image descriptions being independent of exact pixel positions. Signature-based loop-closure detection is closely related to characterizing places by signatures as it is the case for purely topological maps or topo-metric maps with signature-based a signature-based representation of places (sections 3.6.3.2 and 3.6.4.1). Loop-closure detection needs to be a purely vision-based because the robot's position estimate can drift over time from the robot's true position.

For our application, loop closures can occur at the borders of neighboring cleaning segments (section 3.2.3.3). While traveling along its current cleaning lane and approaching an already cleaned segment, the robot is supposed to stop exactly at the border of this segment. Otherwise, uncleaned areas remain or repeated coverage occurs, and the resulting map of the environment becomes inconsistent. Besides that, loop-closure detection can be applied in order to find shortcuts between neighboring cleaning segments.

6.2. Methods

For detecting loop closures based on signatures, we pursue the following approach (figure 6.1): the robot's current camera image $\boldsymbol{C}_p$ is unfolded to a cylindrical image $\boldsymbol{I}_p$, which is used to transform the image into a lower-dimensional image signature $\boldsymbol{p}$ by applying a signature function $\mathrm{s}(\boldsymbol{I}_p)$. As signature functions, we apply functions deriving parameters based on gray-value histograms, on statistical image properties and on absolute Fourier coefficients (section 6.2.1). In case the robot's current position is added as a place node to the dense topo-metric map (section 4.2), the signature $\boldsymbol{p}$ and the panoramic image $\boldsymbol{I}_p$ are attached to that node. As the stored signature can be reused in latter processing steps, repeated evaluations of the signature function are avoided.

In a second step, the current signature $\boldsymbol{p}$ is compared to signatures $\boldsymbol{q}_i$ stored in the topological map. For our particular application, it is sufficient to only compare the snapshots at the borders of already cleaned areas; typical loop-closure applications require to compare the current signature $\boldsymbol{p}$ with all the signatures $\boldsymbol{q}_i$ stored in the map. Comparisons between signatures $\boldsymbol{p}$ and $\boldsymbol{q}$ are computed by applying a dissimilarity function $\mathrm{d}(\boldsymbol{p}, \boldsymbol{q}_i)$ yielding a scalar measure $\hat{\ell}$ of the image dissimilarity. We refer to the comparison function as *dissimilarity* function and not as *distance* function to emphasize the difference between distances in the signature space and spatial distances. As dissimilarity functions we apply standard norm functions as well as histogram-specific dissimilarity functions (section 6.2.2). Based on $\hat{\ell}$ and a threshold ℓ_t, a decision is made whether the images $\boldsymbol{I}_p$ and $\boldsymbol{I}_{q_i}$, and therefore the positions of their acquisition, are identical or not.

6.2.1. Signature Functions

Signature functions s are used to transform the image $\boldsymbol{I}_p$ taken at position $\boldsymbol{x}$ into a global image descriptor $\boldsymbol{p}$. In an ideal case, the resulting signature $\boldsymbol{p} = \mathrm{s}(\boldsymbol{I}_p)$ and therefore also the signature functions s exhibit the following properties:

$$\mathrm{s}\left(\boldsymbol{I}(\boldsymbol{x})\right) = \mathrm{s}\left(\boldsymbol{I}'(\boldsymbol{x}')\right) \text{ iff } \boldsymbol{x} = \boldsymbol{x}' \tag{6.1}$$

$$\mathrm{s}\left(\boldsymbol{I}(x, y, \theta)\right) = \mathrm{s}\left(\boldsymbol{I}(x, y, \theta')\right) \tag{6.2}$$

$$\mathrm{s}\left(\boldsymbol{I}(\boldsymbol{x}, t)\right) = \mathrm{s}\left(\boldsymbol{I}(\boldsymbol{x}, t')\right) \tag{6.3}$$

Equation (6.1) states that —in an ideal case— identical signatures are computed if and only if the images were taken at identical positions in space. This property enforces robustness against perceptual aliasing[1] (section 3.2.3.1), i.e. the ability to distinguish two different positions $\boldsymbol{x}$ and $\boldsymbol{x}'$ with the same visual appearance. Invariance against rotations of the robot is expressed by equation (6.2). Rotational invariance can be achieved by applying signature functions which derive image signatures independent of exact pixel positions. Thus, two images can be recognized as being identical without aligning them w.r.t. a common frame of reference as it is required for the holistic loop-closure detection methods proposed in chapter 5. In contrast to rotational invariance, achieving invariance against perceptual variability (section 3.2.3.2) as is required by equation (6.3) is more difficult. Perceptual variability means that the visual appearance of an image changes due to illumination changes or due to dynamic changes of the scene. Most vision-based navigation strategies are not fully invariant against perceptual variability, but only achieve tolerance up to a certain extent of variability. We expect that some of the combinations of signature functions s and dissimilarity functions d tested in this chapter exhibit a sufficient tolerance against illumination changes to reliably detect loop closures. We therefore avoid the computational effort for additional image preprocessing prior to loop-closure detection as pursued for holistic loop-closure detection methods described in chapter 5.

As signature functions, we have chosen signatures which are (i) rotationally invariant and (ii) can be efficiently computed even with the restricted hardware available on an autonomous cleaning robot. We therefore use signatures based on gray-value histograms (`hist` and `chist`), on image statistics (`cog`, `mm`, `mv`, `ms`, `mk`, `mmv`, `msk`, `mmvs`, `mvsk`, and `mmvsk`), and on Fourier coefficients (`afc` and `zafc`). A categorization of the signatures is depicted in figure 6.2.

Signature Functions Based on Gray-Value Histograms

The signature functions

$$\mathrm{s}_{\texttt{hist}}(\boldsymbol{I}) = \boldsymbol{h}(\boldsymbol{I}) = (h_0, h_1, \ldots, h_{b-1})^\top \text{ and} \tag{6.4}$$

$$\mathrm{s}_{\texttt{chist}}(\boldsymbol{I}) = \boldsymbol{c}(\boldsymbol{I}) = (c_0, c_1, \ldots, c_{b-1})^\top \text{ with } c_k = \sum_{l=0}^{k} h_l \ (k = 0, 1, \ldots, b-1) \tag{6.5}$$

[1]"Perceptual aliasing" is often referred to as "spatial aliasing".

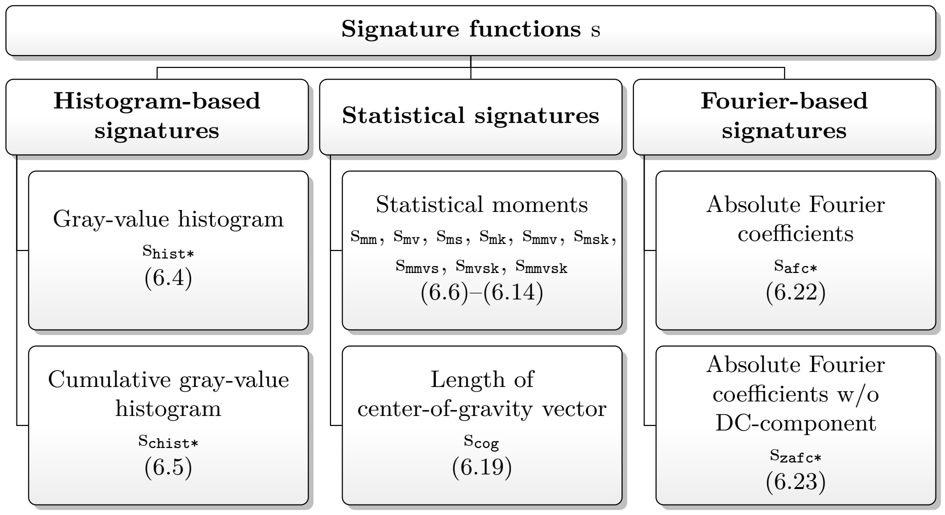

Signature functions s
Histogram-based signatures
Statistical signatures
Fourier-based signatures
Gray-value histogram
$s_{\texttt{hist*}}$
(6.4)
Cumulative gray-value histogram
$s_{\texttt{chist*}}$
(6.5)
Statistical moments
$s_{\texttt{mm}}$, $s_{\texttt{mv}}$, $s_{\texttt{ms}}$, $s_{\texttt{mk}}$, $s_{\texttt{mmv}}$, $s_{\texttt{msk}}$, $s_{\texttt{mmvs}}$, $s_{\texttt{mvsk}}$, $s_{\texttt{mmvsk}}$
(6.6)–(6.14)
Length of center-of-gravity vector
$s_{\texttt{cog}}$
(6.19)
Absolute Fourier coefficients
$s_{\texttt{afc*}}$
(6.22)
Absolute Fourier coefficients w/o DC-component
$s_{\texttt{zafc*}}$
(6.23)

Figure 6.2.: Categorization of signature functions used for signature-based loop-closure detection. The dimensionality of the signature functions marked by a star (∗) not only depends on the number r of rings but also on the number b of histogram bins or Fourier coefficients. The numbers in parentheses refer to the equation defining the signature function.

use the *histogram* $\boldsymbol{h}(\boldsymbol{I})$ and the *cumulative histogram* $\boldsymbol{c}(\boldsymbol{I})$ of relative gray-value frequencies as signatures (e.g. [132, 242, 476, 549, 647, 688] and sections 3.3.1.1, 3.6.3.2 and 3.6.4.1). Both types of histograms contain b bins and are normalized in order to sum up to unity (i.e. $\sum_{k=0}^{b-1} h_k = 1$ and $\sum_{k=0}^{b-1} c_k = 1$). Especially if a small number b of bins is used, we expect histogram-based signatures to be tolerant against smaller illumination changes; stronger changes should be compensated for by histogram-comparison functions with cross-bin matching (section 6.2.2.1).

Signature Functions Based on Image Statistics

Among the signatures based on image statistics, we apply two different groups, namely signatures based on *statistical moments* and the *length of the center of gravity vector*. The first group extends the signatures used by [177, 537] and combine the first four empirical moments mean, variance, skewness, and kurtosis of the image $\boldsymbol{I}$ (sized $w \times h$ pixels) to image signatures. In particular the signatures which are independent of the mean of image intensities (equation (6.15)) should be robust against illumination changes influencing the average image brightness. In detail, we use the following nine signatures:

$$s_{\texttt{mm}}(\boldsymbol{I}) = \operatorname{mean}(\boldsymbol{I}), \tag{6.6}$$
$$s_{\texttt{mv}}(\boldsymbol{I}) = \operatorname{var}(\boldsymbol{I}), \tag{6.7}$$
$$s_{\texttt{ms}}(\boldsymbol{I}) = \operatorname{skew}(\boldsymbol{I}), \tag{6.8}$$
$$s_{\texttt{mk}}(\boldsymbol{I}) = \operatorname{kurt}(\boldsymbol{I}), \tag{6.9}$$
$$s_{\texttt{mmv}}(\boldsymbol{I}) = (\operatorname{mean}(\boldsymbol{I}), \operatorname{var}(\boldsymbol{I}))^\top, \tag{6.10}$$
$$s_{\texttt{msk}}(\boldsymbol{I}) = (\operatorname{skew}(\boldsymbol{I}), \operatorname{kurt}(\boldsymbol{I}))^\top, \tag{6.11}$$
$$s_{\texttt{mmvs}}(\boldsymbol{I}) = (\operatorname{mean}(\boldsymbol{I}), \operatorname{var}(\boldsymbol{I}), \operatorname{skew}(\boldsymbol{I}))^\top, \tag{6.12}$$
$$s_{\texttt{mvsk}}(\boldsymbol{I}) = (\operatorname{var}(\boldsymbol{I}), \operatorname{skew}(\boldsymbol{I}), \operatorname{kurt}(\boldsymbol{I}))^\top, \text{ and} \tag{6.13}$$
$$s_{\texttt{mmvsk}}(\boldsymbol{I}) = (\operatorname{mean}(\boldsymbol{I}), \operatorname{var}(\boldsymbol{I}), \operatorname{skew}(\boldsymbol{I}), \operatorname{kurt}(\boldsymbol{I}))^\top \tag{6.14}$$

with

$$\mathrm{mean}(\boldsymbol{I}) = \frac{1}{wh} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y), \tag{6.15}$$

$$\mathrm{var}(\boldsymbol{I}) = \frac{1}{wh-1} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} (I(x,y) - \mathrm{mean}(\boldsymbol{I}))^2, \tag{6.16}$$

$$\mathrm{skew}(\boldsymbol{I}) = \frac{1}{wh \cdot \mathrm{var}(\boldsymbol{I})^{\frac{3}{2}}} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} (I(x,y) - \mathrm{mean}(\boldsymbol{I}))^3, \text{ and} \tag{6.17}$$

$$\mathrm{kurt}(\boldsymbol{I}) = \frac{1}{wh \cdot \mathrm{var}(\boldsymbol{I})^2 - 3} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} (I(x,y) - \mathrm{mean}(\boldsymbol{I}))^4 \tag{6.18}$$

The naming of these signatures (equations (6.6) to (6.14)) is as follows: the first letter is a `m` indicating that the signature belongs to the class of moment-based signatures, and the following letters correspond to the first letters of the statistical moments (equations (6.15) to (6.18)) used for the signatures.

The second group of statistical signatures is formed by a one-dimensional signature `cog`, which relies on the length of the center of gravity (CoG) position vector. A similar signature was also tested by [177, 537]. The signature is defined as

$$s_{\mathtt{cog}}(\boldsymbol{I}) = \left\| \frac{1}{n} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y) \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} \right\| \text{ with} \tag{6.19}$$

$$n = \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y) \text{ and} \tag{6.20}$$

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = y \begin{pmatrix} \cos\left(\frac{2\pi x}{w}\right) \\ \sin\left(\frac{2\pi x}{w}\right) \end{pmatrix}. \tag{6.21}$$

If image intensities $I(x,y)$ were in equation (6.19) multiplied with Cartesian image coordinates $(x,y)^\top$ rather than with $(\tilde{x},\tilde{y})^\top$ as defined by equation (6.21), the length of the CoG position vector would depend on the robot's orientation (figure 6.3.1). Rotational invariance of the signature is achieved because the length of $(\tilde{x},\tilde{y})^\top$ does not depend on the robot's orientation. This is similar to a polar representation of the panoramic image with the angular coordinate φ depending on the image column x and the radius r being equal to the image row y (figure 6.3.2). With such a representation, the *position* of the center of gravity $\boldsymbol{C}$ also depends on the robot's orientation, but the *length* of its position vector is rotationally independent.

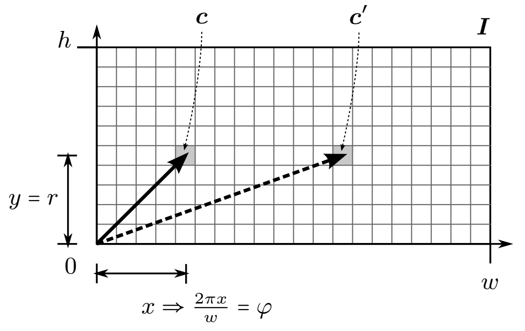
Signature Functions Based on Fourier Coefficients

Image signatures based on absolute Fourier coefficients were first proposed by Menegatti, Maeda, and Ishiguro [421] and were applied to visual homing with parameter models (section 3.5.2.2) by [177, 537]. Instead of applying a 1D FFT to each row of the image $\boldsymbol{I}$ as originally proposed by [421], we average over the columns of the image $\boldsymbol{I}$ in order to obtain a 1D panorama $\bar{\boldsymbol{I}}$, which is then transformed:

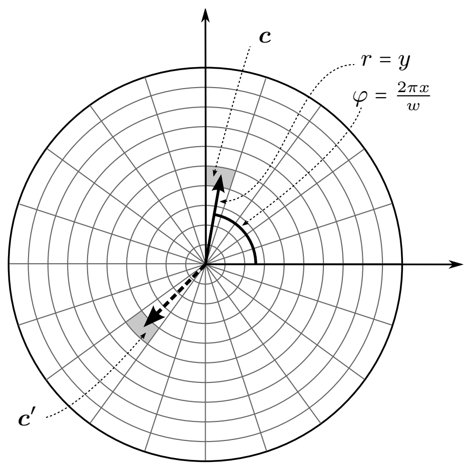
$$s_{\mathtt{afc}}(\boldsymbol{I}) = (f_0, f_1, \dots, f_{b-1})^\top \text{ and} \tag{6.22}$$

$$s_{\mathtt{zafc}}(\boldsymbol{I}) = (f_1, f_2, \dots, f_b)^\top \text{ with} \tag{6.23}$$

$$f_k = \left| \sum_{x=0}^{w-1} \bar{I}(x) \exp\left(-2\pi k i \frac{x}{w}\right) \right|. \tag{6.24}$$



(1) Panoramic image in Cartesian coordinates



(2) Panoramic image in polar coordinates

Figure 6.3.: Computation of the center of gravity (CoG) position vector. The CoG is initially located at position $\boldsymbol{c}$ and moves to position $\boldsymbol{c}'$ after a rotation of the robot. Both, in a Cartesian (1) and in a polar (2) representation of the panoramic image, the *position* of the CoG depends on the robot's orientation. In Cartesian coordinates, the *length* of the position vector pointing towards the CoG (thick arrows) depends on the robot's orientation, whereas it is independent in polar coordinates.

To achieve rotation invariance, the phase information is eliminated by using the absolute values of the Fourier coefficients. The signature `afc` includes the first Fourier coefficient (DC-component) f_0 and is referred to as *absolute Fourier coefficients with DC-component.* For the `zafc` signature, the first Fourier coefficient f_0 is omitted, and the signature is referred to as *absolute Fourier coefficients without DC-component.* Because the first coefficient covers the average image brightness, whereas higher-order coefficients express deviations from the average image brightness, we expect this signature to be more robust against variations of the illumination changing the average image brightness. As most of the image information is contained in the lower-order coefficients (e.g. [421, 495, 588, 611]), both signatures are formed by the first b Fourier coefficients only.

6.2.2. Dissimilarity Functions

Dissimilarity functions d are used to compare two image signatures $\boldsymbol{p} = \mathrm{s}(\boldsymbol{I}_p(\boldsymbol{x}_p))$ and $\boldsymbol{q} = \mathrm{s}(\boldsymbol{I}_q(\boldsymbol{x}_q))$. In an ideal case, the distance functions would exhibit the following properties, which are identical to the properties of image dissimilarity functions described in section 5.2.2:

$$\mathrm{d}(\boldsymbol{p},\boldsymbol{q}) \geq 0, \tag{6.25}$$

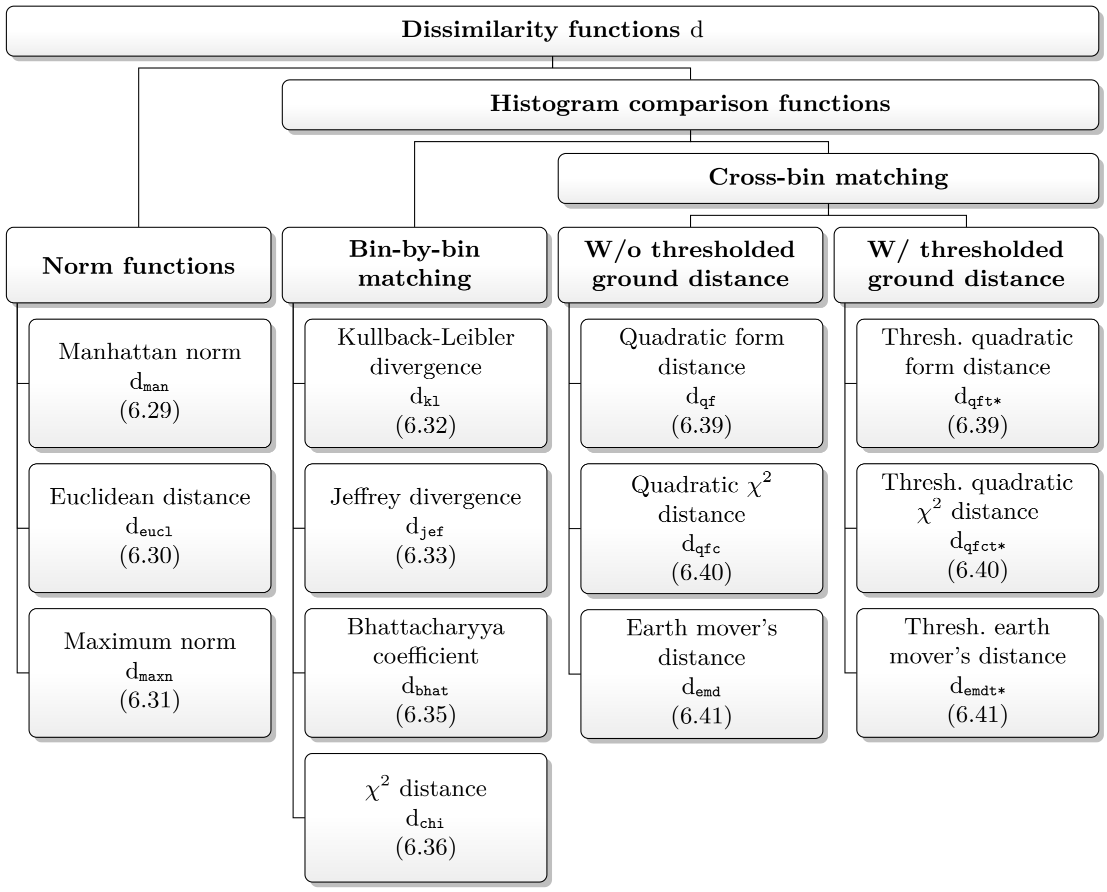
$$\mathrm{d}(\boldsymbol{p},\boldsymbol{q}) = 0 \Leftrightarrow \boldsymbol{p} = \boldsymbol{q}, \tag{6.26}$$

$$\mathrm{d}(\boldsymbol{p},\boldsymbol{q}) = \mathrm{d}(\boldsymbol{q},\boldsymbol{p}), \text{ and} \tag{6.27}$$

$$\mathrm{d}(\boldsymbol{p},\boldsymbol{q}) \overset{\cdot}{\sim} \mathrm{dist}(\boldsymbol{x}_p,\boldsymbol{x}_q). \tag{6.28}$$

Equations (6.25) and (6.26) state that the dissimilarity function d should be zero if and only if the signatures $\boldsymbol{p}$ and $\boldsymbol{q}$ are identical; for other cases d should be positive. Because of the properties of signatures defined in equations (6.1) to (6.3), zero dissimilarity between signatures $\boldsymbol{p}$ and $\boldsymbol{q}$ should only occur if the signatures result from images acquired at identical positions in space. Equation (6.27) requires the dissimilarity function d to be symmetric, and equation (6.28) states that the image dissimilarity $\mathrm{d}(\boldsymbol{p},\boldsymbol{q})$ should depend monotonically on the spatial distance $\mathrm{dist}(\boldsymbol{x}_p,\boldsymbol{x}_q)$ between the positions of image acquisition $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$.

In order to identify promising combinations of signature functions s and dissimilarity functions d, we test a wide range of dissimilarity functions (figure 6.4). The functions can be grouped into norm



Dissimilarity functions d
Histogram comparison functions
Cross-bin matching
Norm functions
Bin-by-bin matching
W/o thresholded ground distance
W/ thresholded ground distance
Manhattan norm $d_{\mathtt{man}}$ (6.29)
Euclidean distance $d_{\mathtt{eucl}}$ (6.30)
Maximum norm $d_{\mathtt{maxn}}$ (6.31)
Kullback-Leibler divergence $d_{\mathtt{kl}}$ (6.32)
Jeffrey divergence $d_{\mathtt{jef}}$ (6.33)
Bhattacharyya coefficient $d_{\mathtt{bhat}}$ (6.35)
χ^2 distance $d_{\mathtt{chi}}$ (6.36)
Quadratic form distance $d_{\mathtt{qf}}$ (6.39)
Quadratic χ^2 distance $d_{\mathtt{qfc}}$ (6.40)
Earth mover's distance $d_{\mathtt{emd}}$ (6.41)
Thresh. quadratic form distance $d_{\mathtt{qft*}}$ (6.39)
Thresh. quadratic χ^2 distance $d_{\mathtt{qfct*}}$ (6.40)
Thresh. earth mover's distance $d_{\mathtt{emdt*}}$ (6.41)

Figure 6.4.: Categorization of dissimilarity functions used to compare global image signatures. Functions tested with multiple parameters are marked by a star (*). The numbers in parentheses refer to the equation defining the dissimilarity function. Cross-bin matching function without thresholded ground distance rely on equation (6.37) as ground distance, whereas their thresholded counterparts apply equation (6.46).

functions ($d_{\mathtt{man}}$, $d_{\mathtt{eucl}}$, and $d_{\mathtt{maxn}}$), histogram-comparison functions with bin-by-bin matching ($d_{\mathtt{kl}}$, $d_{\mathtt{jef}}$, $d_{\mathtt{bhat}}$, and $d_{\mathtt{chi}}$), and histogram-comparison functions with cross-bin matching ($d_{\mathtt{qf}}$, $d_{\mathtt{qfc}}$, $d_{\mathtt{emd}}$, $d_{\mathtt{qft}}$, $d_{\mathtt{qfct}}$, and $d_{\mathtt{emdt}}$).

Norm Functions

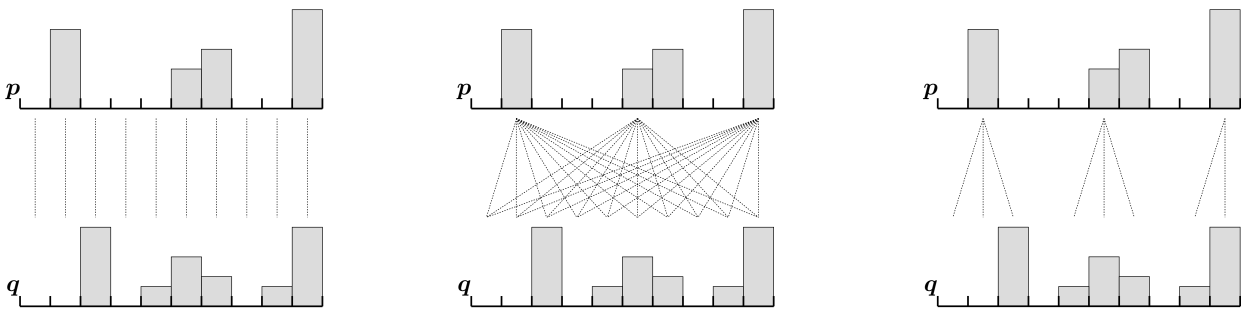
As norm functions we test the *Manhattan norm* ($d_{\mathtt{man}}$), the *Euclidean norm* ($d_{\mathtt{eucl}}$), and the *maximum norm* ($d_{\mathtt{maxn}}$):

$$d_{\mathtt{man}}(\boldsymbol{p},\boldsymbol{q}) = L_1(\boldsymbol{p},\boldsymbol{q}) = \sum_{k=0}^{b-1} |p_k - q_k|, \tag{6.29}$$

$$d_{\mathtt{eucl}}(\boldsymbol{p},\boldsymbol{q}) = L_2(\boldsymbol{p},\boldsymbol{q}) = \sqrt{\sum_{k=0}^{b-1} (p_k - q_k)^2}, \text{ and} \tag{6.30}$$

$$d_{\mathtt{maxn}}(\boldsymbol{p},\boldsymbol{q}) = L_\infty(\boldsymbol{p},\boldsymbol{q}) = \max_{k=0}^{b-1} |p_k - q_k|. \tag{6.31}$$

These functions can be used to compute the dissimilarity for all considered signatures. Norm functions do not exhibit tolerance against illumination changes. Thus, a combination of a signature function and a norm function to compare signatures can only be robust against illumination changes if the signature function tolerates illumination changes.



(1) Bin-by-bin matching **(2)** Cross-bin matching **(3)** Thresholded cross-bin matching

Figure 6.5.: Different types of histogram-comparison functions comparing two histograms $\boldsymbol{p}$ and $\boldsymbol{q}$ by matching their bins p_i and q_j. Matches between bins are indicted by dashed lines. *Bin-by-bin matching* functions only compare corresponding bins (i.e. for $i = j$; subfigure (1)). *Cross-bin matching* functions match between arbitrary bins (i.e. for arbitrary i, j; subfigure (2)). *Thresholded cross-bin matching* functions compare neighboring bins around the corresponding bin (i.e. if $|i - j| < t$ with t being a threshold; subfigure (3)). In case changes of the illumination occur between acquiring the images represented by $\boldsymbol{p}$ and $\boldsymbol{q}$, the image intensity distribution and therefore also the histograms $\boldsymbol{p}$ and $\boldsymbol{q}$ change. Bin-by-bin matching functions are more sensitive against such changes because they only take corresponding bins into account. Cross-bin-matching functions are more tolerant because they compute the similarity between $\boldsymbol{p}$ and $\boldsymbol{q}$ not only from matching corresponding bins but also from matching each bin of $\boldsymbol{p}$ with several (or all) bins of $\boldsymbol{q}$. For sake of a better visibility, only a reduced number of bin matches (dashed lines) is depicted in subfigures (2) and (3); (thresholded) histogram-comparison functions with cross-bin matching of course compare *all* bins of $\boldsymbol{p}$ with the corresponding bins of $\boldsymbol{q}$. After Rubner, Tomasi, and Guibas [549].

Histogram-Comparison Functions With Bin-by-Bin Matching

Histogram comparison functions with bin-by-bin matching only compare corresponding bins of two histograms (figure 6.5.1), i.e. for the histograms $\boldsymbol{p}$ and $\boldsymbol{q}$ they only match the histogram entries p_i and q_i for $i = j$. Bin-by-bin matching functions can be computed efficiently, but they are (i) more sensitive to bin sizes and boundaries, and (ii) less robust against changes of the illumination than cross-bin matching methods (section 6.2.2.1). As bin-by-bin matching functions, we test the *Kullback-Leibler divergence* $\mathrm{d}_{\texttt{kl}}$, the *Jeffrey divergence* $\mathrm{d}_{\texttt{jef}}$, the *Bhattacharyya coefficient* $\mathrm{d}_{\texttt{bhat}}$, and the χ^2 *distance* $\mathrm{d}_{\texttt{chi}}$. The Kullback-Leibler divergence (Kullback [348]) is defined as

$$\mathrm{d}_{\texttt{kl}}(\boldsymbol{p}, \boldsymbol{q}) = \sum_{k=0}^{b-1} p_k \log \frac{p_k}{q_k} \tag{6.32}$$

and measures the distance between the probability distributions represented by the histograms $\boldsymbol{p}$ and $\boldsymbol{q}$. As the Kullback-Leibler divergence is not symmetric and therefore violates equation (6.27), we also test the Jeffrey divergence proposed by Puzicha, Hoffmann, and Buhmann [524]

$$\mathrm{d}_{\texttt{jef}}(\boldsymbol{p}, \boldsymbol{q}) = \sum_{k=0}^{b-1} \left(p_k \log \frac{p_k}{m_k} + q_k \log \frac{q_k}{m_k} \right) \text{ with} \tag{6.33}$$

$$m_k = \frac{p_k + q_k}{2}, \tag{6.34}$$

which is a symmetric variant of the Kullback-Leibler divergence. The negative Bhattacharyya coefficient (Comaniciu, Ramesh, and Meer [117] and Kailath [319])

$$\mathrm{d}_{\texttt{bhat}} = -\log \sum_{k=0}^{b-1} \sqrt{p_k q_k} \tag{6.35}$$

measures the overlap between the probability distributions approximated by the histograms $\boldsymbol{p}$ and $\boldsymbol{q}$. The χ^2 distance function (RUBNER, TOMASI, and GUIBAS [549])

$$\mathrm{d}_{\mathtt{chi}}(\boldsymbol{p},\boldsymbol{q}) = \sum_{k=0}^{b-1} \frac{(p_k - m_k)^2}{m_k} \tag{6.36}$$

(with m_k defined by equation (6.34)) is based on χ^2 statistics and thus computes the likelihood that the image signatures are histograms of different distributions. Like all histogram-comparison functions, the functions $\mathrm{d}_{\mathtt{kl}}$, $\mathrm{d}_{\mathtt{jef}}$, $\mathrm{d}_{\mathtt{bhat}}$, and $\mathrm{d}_{\mathtt{chi}}$ can only be used to compare the histogram-based signatures $\mathrm{s}_{\mathtt{hist}}$ and $\mathrm{s}_{\mathtt{chist}}$.

6.2.2.1. Histogram-Comparison Functions With Cross-Bin Matching

Histogram-comparison functions with cross-bin matching do not only match corresponding bins between two histograms $\boldsymbol{p}$ and $\boldsymbol{q}$, but rather match each bin p_i of histogram $\boldsymbol{p}$ with several or all bins q_j of histogram $\boldsymbol{q}$ (figure 6.5). Comparison functions matching *all* bins are also referred to as *cross-bin matching functions without thresholded ground distance* (figure 6.5.2); functions matching each bin p_i with a small *subset of bins* from $\boldsymbol{q}$ are referred to as *cross-bin matching functions with thresholded ground distance* (figure 6.5.3).

The advantage of cross-bin matching functions over bin-by-bin matching functions is that they are more tolerant against illumination changes. By matching each bin of $\boldsymbol{p}$ with several bins from $\boldsymbol{q}$, such comparison functions can still measure similarities between histograms even if their intensity distribution changed due to variations of the illumination conditions (figure 6.5). However, they are computationally more demanding than bin-by-bin matching functions. Like bin-by-bin matching functions, the cross-bin matching functions described in the following are all restricted to the histogram-based signatures $\mathrm{s}_{\mathtt{hist}}$ and $\mathrm{s}_{\mathtt{chist}}$.

Cross-Bin Matching Functions Without Thresholded Ground Distance

Comparison functions of this class match each bin p_i of histogram $\boldsymbol{p}$ with each bin q_j of histogram $\boldsymbol{q}$. Pairings between bins are weighted according to the similarity of the features which are represented by the bins —in our case image intensities. Pairs of similar or corresponding bins receive high weights, pairs between distant bins (e.g. between bins representing bright and dark image regions) receive low weights. The weights are computed depending on a *ground distance* function $\mathrm{g}(i,j)$, which expresses the similarity between the intensities represented by bins i and j (RUBNER, TOMASI, and GUIBAS [549]). As ground distance, it is sufficient to use the absolute difference

$$\mathrm{g}(i,j) = |i - j| \tag{6.37}$$

between bin numbers rather than a difference measure depending on image intensities (see appendix D.1 for a proof). Based on $\mathrm{g}(i,j)$ the weight $A_{i,j}$ for comparing bin i and j can be computed by

$$A_{i,j} = 1 - \frac{\mathrm{g}(i,j)}{\mathrm{g}_{\max}} \text{ with } \mathrm{g}_{\max} = \max(\mathrm{g}(i,j)). \tag{6.38}$$

The matching functions $\mathrm{d}_{\mathtt{qf}}$ and $\mathrm{d}_{\mathtt{qfc}}$ described in the following combine these weights to a symmetric weight matrix $\boldsymbol{A}$.

Two of the three tested cross-bin matching functions without thresholded ground distance rely on a quadratic form. These are the *quadratic form distance function* $\mathrm{d}_{\mathtt{qf}}$ (NIBLACK et al. [489]) and the *quadratic χ^2 distance function* $\mathrm{d}_{\mathtt{qfc}}$ (Pele and Werman [509]), which combines the χ^2 function $\mathrm{d}_{\mathtt{chi}}$ with the quadratic from distance function $\mathrm{d}_{\mathtt{qf}}$:

$$\mathrm{d}_{\mathtt{qf}}(\boldsymbol{p},\boldsymbol{q}) = \sqrt{(\boldsymbol{p}-\boldsymbol{q})^{\top}\boldsymbol{A}(\boldsymbol{p}-\boldsymbol{q})} \text{ and} \tag{6.39}$$

$$\mathrm{d}_{\mathtt{qfc}}(\boldsymbol{p},\boldsymbol{q}) = \sqrt{\sum_{i=0}^{b-1}\sum_{j=0}^{b-1}\left(\frac{p_i - q_i}{\left(\sum_{k=0}^{b-1}(p_k+q_k)A_{k,i}\right)^n}\right)\left(\frac{p_j - q_j}{\left(\sum_{k=0}^{b-1}(p_k+q_k)A_{k,j}\right)^n}\right)A_{i,j}} \tag{6.40}$$

with n being a normalization constant usually chosen to be $n = 0.9$.

The third comparison function is the *earth-mover's distance* (EMD) originally proposed by Rubner, Tomasi, and Guibas [548, 549], which is often used as reference comparison functions for histogram comparison and image retrieval (e.g. [508, 509]; review: [132]). It formulates the cross-bin matching as a transportation problem, a sub-domain of linear optimization (textbook: [108]). By solving the transportation problem, the minimal cost to transform one histogram into the other is computed:

$$\mathrm{d}_{\mathtt{emd}}(\boldsymbol{p},\boldsymbol{q}) = \min_{F_{ij}} \frac{\sum_{i=1}^{b-1}\sum_{j=1}^{b-1} F_{ij}\,\mathrm{g}(i,j)}{\sum_{i=1}^{b-1}\sum_{j=1}^{b-1} F_{ij}} \tag{6.41}$$

with F_{ij} being the "flow" which is transported between bins p_i and q_j. The optimization is constrained by

$$F_{ij} \geq 0 \tag{6.42}$$

$$\sum_{j=0}^{b-1} F_{ij} \leq p_i, \tag{6.43}$$

$$\sum_{i=0}^{b-1} F_{ij} \leq q_j, \text{ and} \tag{6.44}$$

$$\sum_{i=0}^{b-1}\sum_{j=0}^{b-1} F_{ij} = \min\left(\sum_{i=0}^{b-1} p_i, \sum_{j=0}^{b-1} q_j\right) \tag{6.45}$$

Metaphorically, the EMD represents histograms as heaps of earth and measures the minimal energy required to "shovel" one set of heaps into the other [67]. The constraints define how earth is shoveled: equation (6.42) only allows to move from $\boldsymbol{p}$ to $\boldsymbol{q}$, equations (6.43) and (6.44) limit the amount of earth moved from $\boldsymbol{p}$ to $\boldsymbol{q}$, and equation (6.45) assures that as much earth as possible is moved. For our experiments, we do not rely on the original variant of the EMD as proposed by [548, 549], but apply the Fast-EMD proposed by Pele and Werman [508]. It follows the same principles as the original EMD as outlined above, but is more efficient.

Cross-Bin Matching Functions With Thresholded Ground Distance

Matching arbitrary bins as is the case for cross-bin matching functions *without* thresholded ground distance can increase the probability of mismatches between two histograms. This drawback can be avoided by restricting the cross-bin matches to comparing only bins which represent *similar* image regions. For this purpose, the application of a *thresholded* ground distance

$$\mathrm{g}_t(i,j) = \min(\mathrm{g}(i,j),t). \tag{6.46}$$

was proposed by [508, 509], which can be used as a replacement for equation (6.37) with all three dissimilarity functions defined in equations (6.39) to (6.41). The comparison of two bins p_i and q_j with a ground distance exceeding the threshold t will, according to equation (6.38), receive a zero weight and therefore does not contribute to the overall dissimilarity value computed for the histograms $\boldsymbol{p}$ and $\boldsymbol{q}$. All cross bin matching functions described above are also tested in a variant with thresholded ground distance and are referred to as *thresholded quadratic form distance* $\mathrm{d}_{\texttt{qft}}$, *thresholded quadratic* χ^2 *distance function* $\mathrm{d}_{\texttt{qfct}}$, and *thresholded earth-mover's distance* $\mathrm{d}_{\texttt{emdt}}$, respectively. The distance threshold t is an additional parameter of these functions.

6.2.3. Dimensionality of Signatures

The proposed signatures have fixed dimensionalities ($b = 1$ for $\mathrm{s}_{\texttt{cog}}$, $\mathrm{s}_{\texttt{mm}}$, $\mathrm{s}_{\texttt{mv}}$, $\mathrm{s}_{\texttt{ms}}$, and $\mathrm{s}_{\texttt{mk}}$; $b = 2$ for $\mathrm{s}_{\texttt{mmv}}$, and $\mathrm{s}_{\texttt{msk}}$; $b = 3$ for $\mathrm{s}_{\texttt{mmvs}}$ and $\mathrm{s}_{\texttt{mvsk}}$; $b = 4$ for $\mathrm{s}_{\texttt{mvsk}}$) or dimensionalities b equal to the number of histogram bins ($\mathrm{s}_{\texttt{hist}}$ and $\mathrm{s}_{\texttt{chist}}$) and the number of Fourier coefficients ($\mathrm{s}_{\texttt{afc}}$ and $\mathrm{s}_{\texttt{zafc}}$). The dimensionality of the computed image signature $\boldsymbol{p}$ influences the computational requirements for storing and comparing signatures as well as their discriminability. Finding an appropriate signature dimension is a trade-off because the first aspect favors low-dimensional signatures, whereas a larger number of dimensions should facilitate more reliable loop-closure detection.

In order to increase the dimensionality of the image signatures, we follow the approach of Gonzalez-Barbosa and Lacroix [242] and subdivide the cylindrical image $\boldsymbol{I}$ into r subpanoramas $\boldsymbol{S}_j$ $(0 \leq j \leq r-1)$ of equal height. In the original camera image, each subpanorama $\boldsymbol{S}_j$ corresponds to a concentric ring (figure 5.1). For each of these subimages, an image signature $\boldsymbol{p}_j = \mathrm{s}(\boldsymbol{S}_j)$ is computed, and the resulting signatures are combined to the signature

$$\boldsymbol{p} = \left(\boldsymbol{p}_0^\top, \boldsymbol{p}_1^\top, \dots, \boldsymbol{p}_{r-1}^\top\right)^\top . \tag{6.47}$$

Hence, the overall dimension of the resulting signature $\boldsymbol{p}$ is the product of the sub-signature's dimensionality b and the number of subimages r.

For $r > 1$, we then compute the dissimilarity $\hat{\ell}$ of two image signatures $\boldsymbol{p}$ and $\boldsymbol{q}$ by summing up the dissimilarities of their sub-signatures:

$$\mathrm{d}(\boldsymbol{p}, \boldsymbol{q}) = \sum_{j=0}^{r-1} \mathrm{d}\left(\boldsymbol{p}_j, \boldsymbol{q}_j\right) . \tag{6.48}$$

By treating the parameter vector not as a single vector but rather as a set of sub-signatures, this ensures that the cross-bin matching functions do not match between sub-histograms obtained for different rings.

6.2.4. Robustness Against Changes of the Illumination

Robustness against changes of the illumination is essential because changes of the illumination conditions can considerably alter the appearance of the perceived images (section 3.2.3.2). For signature-based loop-closure detection, robustness against changes of the illumination can be obtained by an appropriate combination of a signature function s and a dissimilarity function d. To theoretically analyze the robustness of the considered combinations of signature and dissimilarity functions against changes of the illumination, we follow the approach described in section 5.2.4. As a first step, we analyze how the resulting signature $\boldsymbol{p}'$ is influenced if the signature function s is applied to an image undergoing a linear transformation of pixel intensities:

$$\boldsymbol{p}' = \mathrm{s}(a\boldsymbol{I} + o). \tag{6.49}$$

The results of this analysis are summarized in table 6.1; for a detailed derivation the reader is referred to appendix D.2. The results show that both the scale a and the offset o of the intensity transformation are only compensated for by the statistical signature functions $\mathrm{s}_{\texttt{ms}}$, $\mathrm{s}_{\texttt{mk}}$, and $\mathrm{s}_{\texttt{msk}}$. The functions $\mathrm{s}_{\texttt{mv}}$, $\mathrm{s}_{\texttt{mvsk}}$, and $\mathrm{s}_{\texttt{zafc}}$ can compensate for shifts o of the image brightness, but cannot

Table 6.1.: Robustness of signature functions s against changes of the illumination modeled by a linear intensity transformation of the input image. If the parameters a or o of the intensity transformation $a\boldsymbol{I} + o$ are compensated for by the signature function s, the corresponding cell is marked by ✓; otherwise it is marked by ✗.

Parameter	Signature function						
	$s_{\texttt{hist}}$	$s_{\texttt{mm}}$	$s_{\texttt{mv}}$	$s_{\texttt{ms}}$	$s_{\texttt{cog}}$	$s_{\texttt{afc}}$	$s_{\texttt{zafc}}$
	$s_{\texttt{chist}}$	$s_{\texttt{mmv}}$	$s_{\texttt{mvsk}}$	$s_{\texttt{mk}}$			
		$s_{\texttt{mmvs}}$		$s_{\texttt{msk}}$			
		$s_{\texttt{mmvsk}}$					
Scale a	—[1]	✗	✗	✓	✗[2]	✗	✗
Offset o	—	✗	✓	✓	✗	✗	✓

[1] For small intensity changes (i.e. $a \approx 1$ and $o \approx 0$) and a small number of bins it is likely that the signatures can compensate for changes of the illumination.

[2] In case the image intensities are only scaled (i.e. $o = 0$), the scale a is compensated: $s(a\boldsymbol{P}) = s(\boldsymbol{P})$.

compensate for intensity scalings. In the special case of a scaling of the input image without a constant additional change of the image intensities (i.e. $o = 0$), the signature function $s_{\texttt{cog}}$ can compensate for the scaling a. Due to the linearity of the mean and the Fourier transformation, the signature functions $s_{\texttt{mm}}$, $s_{\texttt{mmv}}$, $s_{\texttt{mmvs}}$, $s_{\texttt{mmvsk}}$, and $s_{\texttt{afc}}$ cannot compensate for the linear intensity transformation. The influence of the intensity transformation onto the histogram-based signatures $s_{\texttt{hist}}$ and $s_{\texttt{chist}}$ is difficult to predict: on the one hand side, the histograms approximate the intensity distributions of the images. Hence changes of the illumination will effect the resulting image histogram. On the other hand side, the image histogram strongly depends on the size and the limits of the bins. For this reason, it is likely that the histogram-based signatures are robust against changes of the illumination if the changes are small and the histogram bins are large (i.e. for small b).

The norm functions $d_{\texttt{man}}$, $d_{\texttt{eucl}}$, and $d_{\texttt{maxn}}$ rely on differences of the signature's components. Thus, they cannot compensate for changes of the illumination resulting in a change of the signature. The same holds for bin-by-bin matching functions $d_{\texttt{kl}}$, $d_{\texttt{jef}}$, $d_{\texttt{bhat}}$, and $d_{\texttt{chi}}$ for comparing histograms. For these two types of dissimilarity functions, the signature function is the only possibility to obtain tolerance against changes of the illumination. Cross-bin matching functions were developed to be more robust against variations of the image intensities as they can be caused by illumination changes than the bin-by-bin matching functions (e.g. [509, 549]). The better tolerance of these methods results from not only matching corresponding bins but also neighboring ($d_{\texttt{qft}}$, $d_{\texttt{qfct}}$, and $d_{\texttt{emdt}}$) or even distant bins ($d_{\texttt{qf}}$, $d_{\texttt{qfc}}$, and $d_{\texttt{emd}}$). We expect the cross-bin matching functions to be sufficiently robust to allow for reliable loop-closure detection under realistic illumination conditions. For this reason, we do not consider image preprocessing techniques as we did for holistic loop-closure detection methods (chapter 5).

6.3. Database Experiments

As database experiments allow for rapidly screening a large number of combinations of signature and dissimilarity functions, database experiments were performed to identify promising combinations and to optimize the involved parameters. Besides that, we will assess the method's robustness against changes of the illumination and computational aspects. The best combinations identified in this chapter were additionally tested in real-robot experiments in order to assess their performance under more realistic conditions (section 6.4). The remainder of the section is structured as follows: the experimental procedure is described in section 6.3.1. Results are presented in section 6.3.2 and finally discussed in section 6.3.3.

6.3.1. Methods

To assess the performance of the methods proposed in section 6.2, the methods were implemented (section 6.3.1.1) and systematically tested (section 6.3.1.2). The obtained results were then evaluated according to the procedure described in section 6.3.1.3.

6.3.1.1. Implementation

The described methods were implemented in the software framework developed at the Bielefeld Computer Engineering Group, which internally relies on the FFTW library [I27] for computing the Fourier transformation. As the implementation was done by rapid prototyping, possibilities for code optimization were not considered. The image unfolding routines as well as the functions computing image signatures from the subimages were implemented in C/C++. The fusion of sub-signatures to an image signature was programmed in Tcl with image signatures being represented as lists of lists. Computing the signature dissimilarity according to equation (6.48) is also implemented in Tcl. For each sub-signature, the dissimilarity is computed by calling functions of the linear algebra package contained in the Tcl standard library [I93], by using the C/C++ matrix library developed at the Computer Engineering Group or —in case of the EMD– by reusing C/C++ code downloadable from the internet [I76]. The software for data evaluation was written by Lorenz Hillen as Matlab code (Release 14, Service Pack 2) in combination with a C/C++program for computation of the AUC values. This program was provided by Oliver Schlüter and was developed during the course of his bachelor's project [575].

6.3.1.2. Experimental Procedure

In addition to testing several combinations of signature functions s and dissimilarity functions d, we systematically varied the number of subimages r and —for the histogram-based and the Fourier-based signatures— the sub-signature dimensionality b according to

$$r \in \{1, 2, 4, 8, 16, 32\} \text{ and} \tag{6.50}$$

$$b \in \{1, 2, 4, 8, 16, 32\}. \tag{6.51}$$

These parameter combinations resulted in signatures with an overall dimensionality $r \cdot b$ ranging from 1 to 1024. The distance threshold t of the histogram-comparison functions with thresholded ground distance was varied in

$$t \in \{2, 4\}. \tag{6.52}$$

These parameter variations result in a total of 1548 combinations. We perform such exhaustive experiments (i) to find the best combination of signature function s, dissimilarity function d, and parameters for our requirements, (ii) to achieve a fair comparison between different methods, and (iii) to not preassume certain combinations to be more promising or better suited than others.

In order to obtain results which are comparable to the results of the experiments for loop-closure detection based on global image comparisons (chapter 5), we use the same databases [S1] as used for our experiments on holistic loop-closure detection (chapter 5) containing images sized 461×64 pixels; please refer to section 5.3.1.1 and appendix B for details on these databases. For each database, $(n_x n_y)^2$ dissimilarity values

$$\hat{\ell}_{i,j} = \mathrm{d}\left(\mathrm{s}(\boldsymbol{I}_i), \mathrm{s}(\boldsymbol{I}'_j)\right) \tag{6.53}$$

were computed by using each of the $n_x n_y$ images as snapshot $\boldsymbol{I}_i$ $(0 \le i < n_x n_y)$ and comparing the resulting signature $\mathrm{s}(\boldsymbol{I}_i)$ to all signatures $\mathrm{s}(\boldsymbol{I}'_j)$ $(0 \le j < n_x n_y)$ of the same database (with n_x and n_y being the number of snapshots contained in the current database in x- and y-direction, respectively).

Table 6.2.: Categorization of AUC values for analyzing loop-closure detection performance. See also section 5.3.2.1; the table is identical to table 5.3.

Category	Range of AUC values	Description
(− − −)	$0.00 \leq \text{AUC} < 0.60$	Loop-closure detection not possible
(∘)	$0.60 \leq \text{AUC} < 0.90$	Moderate loop-closure detection performance
(+)	$0.90 \leq \text{AUC} < 0.98$	Good performance
(++)	$0.98 \leq \text{AUC} < 1.00$	Very good performance
(+ + +)	$\text{AUC} = 1.0$	Perfect classification

As in section 5.3.1.1, the second image $\boldsymbol{I}'_j$ is rotated and disturbed by Gaussian noise of standard deviation $\sigma = 0.05$. The effects of this disturbance are visualized in figure 5.6.

6.3.1.3. Evaluation

The results obtained by the experimental procedure described in section 6.3.1.2 were grouped into two *test groups* referred to as `day|day` group and `day|night` group. The first group subsumes all the databases collected during day under natural illumination conditions. The `day|night` group pools the results of cross-database experiments pairing images acquired at identical positions in space but at different points in time. This allows for assessing the performance of the proposed methods under strong illumination changes. As the images contained in the `day|night` databases were acquired once during the day under natural illumination conditions and once during the night under artificial illumination conditions. Both for the daytime and the nighttime images, the illumination conditions were nearly constant. The resulting changes of the illumination are very strong, but only reflect a subset of the variety of illumination conditions which can occur during real-world experiments. The real-robot experiments (section 6.4) will test the proposed methods under more realistic illumination conditions. Example images from the databases and their positions of image acquisitions are shown in figures B.1 to B.4; a more detailed description of the databases is given in section 5.3.1.1.

Loop-Closure Detection Performance

Receiver operator characteristics (ROC) are a standard method for evaluating the performance of a classifier based on its *true-positive* rate (TPR) and its *false-negative* rate (FNR). The resulting ROC curves are often analyzed graphically (figure 5.9), but a graphical evaluation is in our case not tractable because of the large number of tested parameter combinations. We rather compute the area under the curve (AUC), which gives a scalar measure with 1, 0.5, and zero indicating perfect classification, chance level, and complete misclassification, respectively. For a further discussion of the AUC measure, the reader is referred to section 5.3.2.1 or to the relevant literature (e.g. [168, 275, 523], textbook: [148]).

The parameter combinations defined by equations (6.50) and (6.51) result in a total of 1548 different methods, i.e. different combinations of signature functions s, dissimilarity functions d, and different combinations of the parameters r, b, and t. For each different method, two AUC values are computed, one for the `day|day` and one for the `day|night` group.

The main difficulty for evaluating the loop-closure detection performance is to get an overall impression of the performance and to identify a small number of methods with a best possible performance for *both* test groups. For this purpose, we categorize the AUC value for each test group into five different bins as defined by table 6.2. These categories allow to quickly inspect the performance of a set of measures (e.g. all methods of the `day|day` group with (i) Fourier-based signatures comparing, (ii) the Euclidean distance to compare images, and (iii) arbitrary dimensionality) by computing a histogram that visualizes the proportions of the AUC values belonging to each category. In order to identify promising combinations for *both* test groups, one can compute 2D histograms representing e.g. the proportion of methods with perfect categorization (+ + +) for the `day|day` group and very good performance (+) for the `day|night` group. In an ideal case, the method would achieve perfect classification (+ + +) for both test groups. With our particular data, for which most of the 25 possible combinations do not occur, it turned out to be sufficient to identify the most promising combinations w.r.t. loop-closure detection from two 1D histograms (one for each test group). We therefore do not analyze the joint 2D distributions of the AUC values computed for each test group, but only the two 1D marginal distributions. By this means, we can represent a large number of AUC values by a compact and easy to analyze representation. This representation allows us to identify a small number of promising combinations to be further analyzed in more detail and to exclude a large number of methods which do not allow for robust loop-closure detection. For these combinations, we will then analyze how the AUC values depend on the choice of the method's parameters r and b. The results of the first and second evaluation step are described in sections 6.3.2.1 and 6.3.2.2, respectively.

Robustness Against Perceptual Aliasing and Perceptual Variability

As a further evaluation, we analyze the robustness against perceptual aliasing and perceptual variability of the proposed methods. Perceptual aliasing and perceptual variability influence the visual appearance of the images and can cause the loop-closure detection methods to fail. In this case, a pair of images acquired at different positions in space appears to be more similar than the two images acquired at identical position in space (section 5.3.2.2 and figure 5.10). For our image-database experiments, we can evaluate how often such situations occur and can —in case of mismatches— compute the distance between the position of the expected match and the computed match. We therefore follow the procedure described in section 5.3.2.2 and compute the percentage $P_{\texttt{corr}}$ of correct matches, $P_{\texttt{misc}}$ of mismatches in the direct vicinity of the expected match and $P_{\texttt{misf}}$ of distant mismatches. Furthermore, we analyze the mean spatial distance dist_{avg} and median spatial distances dist_{med} between the computed match and the expected match. These measures allow us to draw conclusions, how often mismatches occur and how strongly these mismatches influence the robot's navigation capabilities. The results of this evaluation are summarized in section 6.3.2.3.

Computation Time of Signature and Dissimilarity Functions

To measure the computation time required to derive and compare signatures, we follow the procedure described in section 5.3.2.4. For measuring the time required to compute the signatures, the median over 1000 repetitions is computed for each signature function and for each parameter combination of r and b. For profiling dissimilarity functions, we compute the median over 1000 repetitions of measuring for each combination of r and b the time required to compare two signatures. Because comparing signatures only depends on the signature's dimensionality but not on the signature function, the computing times were only measured for comparing histogram-based signatures. Results are presented in sections 6.3.2.4 and 6.3.2.5 for signature and dissimilarity functions, respectively.

6.3.2. Results

This section describes the results of evaluating the proposed methods with respect to loop-closure detection performance (sections 6.3.2.1 and 6.3.2.2), perceptual aliasing and perceptual variability (section 6.3.2.3), and computing times (sections 6.3.2.4 and 6.3.2.5). With the parameters as defined in section 6.3.1.2, a total of 1548 methods, i.e. different combinations of signature and comparison functions, are tested. Each method is first tested with two *test* groups, namely the `day|day` and the `day|night` groups, containing images acquired under natural illumination conditions and simulating strong illumination changes (section 6.3.1.3), and later on evaluated with two *evaluation methods* yielding a scalar (performance of loop-closure detection) and five (robustness against perceptual aliasing and perceptual variability) performance measures. An extensive description and discussion of the obtained results is therefore far beyond the scope of this dissertation. We rather divide the obtained data into six *evaluation groups*, which result from the categorizations of signatures and dissimilarity functions depicted in figures 6.2 and 6.4:

`four-norm`:	Fourier-based signatures compared with norm functions (i.e. $s_{\texttt{afc}}$ and $s_{\texttt{zafc}}$ with $d_{\texttt{man}}$, $d_{\texttt{eucl}}$, and $d_{\texttt{maxn}}$),
`stat-norm`:	statistical signatures compared with norm functions (i.e. $s_{\texttt{mm}}$, $s_{\texttt{mv}}$, $s_{\texttt{ms}}$, $s_{\texttt{mk}}$, $s_{\texttt{mmv}}$, $s_{\texttt{msk}}$, $s_{\texttt{mmvs}}$, $s_{\texttt{mvsk}}$, and $s_{\texttt{mmvsk}}$ with $d_{\texttt{man}}$, $d_{\texttt{eucl}}$, and $d_{\texttt{maxn}}$),
`hist-norm`:	histogram-based signatures compared with norm functions (i.e. $s_{\texttt{hist}}$ and $s_{\texttt{chist}}$ with $d_{\texttt{man}}$, $d_{\texttt{eucl}}$, and $d_{\texttt{maxn}}$),
`hist-bin`:	histogram-based signatures compared with bin-by-bin matching functions (i.e. $s_{\texttt{hist}}$ and $s_{\texttt{chist}}$ with $d_{\texttt{kl}}$, $d_{\texttt{jef}}$, $d_{\texttt{bhat}}$, and $d_{\texttt{chi}}$),
`hist-cbin`:	histogram-based signatures compared without thresholded cross-bin matching (i.e. $s_{\texttt{hist}}$ and $s_{\texttt{chist}}$ with $d_{\texttt{qf}}$, $d_{\texttt{qfc}}$, and $d_{\texttt{emd}}$), and
`hist-tcbin`:	histogram-based signatures compared with thresholded cross-bin matching (i.e. $s_{\texttt{hist}}$ and $s_{\texttt{chist}}$ with $d_{\texttt{qft}}$, $d_{\texttt{qfct}}$, and $d_{\texttt{emdt}}$).

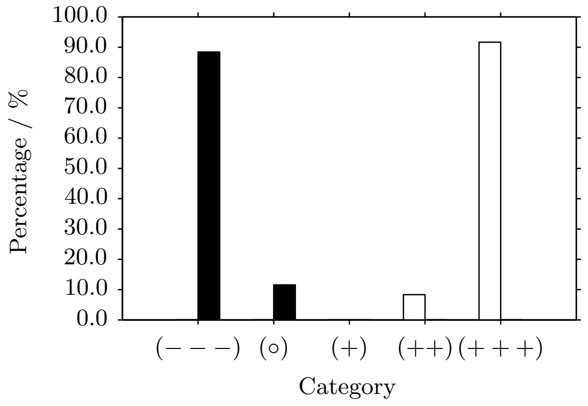
Within these groups, we pool over the parameters r, b, and —for histogram-comparison functions with cross-bin matching— t. By this means, we reduce the entire data to a relevant subset allowing us (i) to identify the most promising methods and (ii) to compare these methods between the different *evaluation* and *test* groups. This procedure is applied in sections 6.3.2.1 and 6.3.2.3.

6.3.2.1. Coarse Screening of Loop-Closure Detection Performance

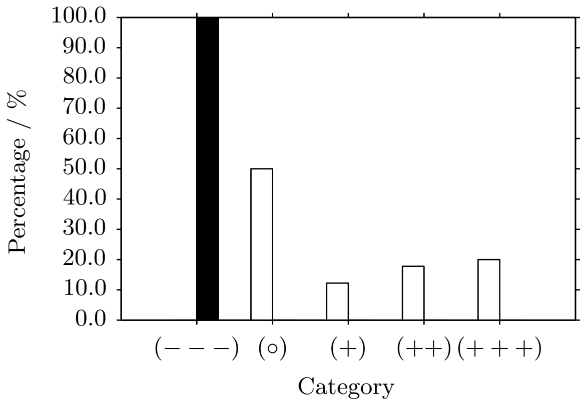
To identify promising combinations of signature functions s and dissimilarity functions d, the AUC values computed for the `day|day` and `day|night` *test* groups were (i) binned according table 6.2 and frequency distributions were computed as described in section 6.3.1.3 and (ii) pooled into six *evaluation* groups (`four-norm`, `stat-norm`, `hist-norm`, `hist-bin`, `hist-cbin`, and `hist-tcbin`; section 6.3.2). The results are depicted in figure 6.6 with one subfigure for each *evaluation* group, and white and black bars depicting the distribution of the `day|day` and the `day|night` *test* group, respectively. The overall impression of the results is that for the `day|day` group many methods exist which achieve perfect (+ + +) or very good classification (++). This contrasts the results for the `day|night` group: there, all methods fall into the categories (− − −) or (∘) for which classification is either not possible or only with moderate results. In the following, we will briefly describe the results for the six evaluation groups.

For comparing Fourier-based signatures by norm functions (`four-norm`; figure 6.6.1), the best possible combination can fall into category (+ + +) for the `day|day` group and in category (∘) for the `day|night` group. The category contains 25 different methods (table 6.3) with AUC values for the `day|night` group between 0.600 and 0.612. All combinations rely on the $s_{\texttt{afc}}$ signature, and for comparing signatures, 2 of 25 use the L_1 norm $d_{\texttt{man}}$, 7 the Euclidean norm $d_{\texttt{eucl}}$ and 16 the maximum norm $d_{\texttt{maxn}}$. The best performance is obtained for $s_{\texttt{afc}}$, $d_{\texttt{maxn}}$, $r = 32$ and $b \in \{8, 16, 32\}$ and achieves an AUC value of 0.612.

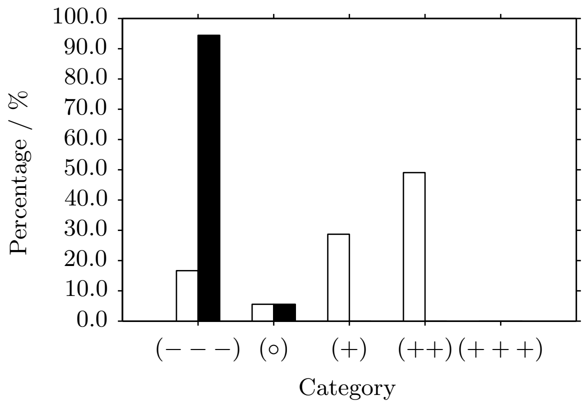
The group of methods containing statistical signatures compared by norm functions (`stat-norm`) achieves a broad distributions of AUC values for the `day|day` group with categories (++) and (+ + +) each accounting for approximately 20 % (figure 6.6.2). However, all AUC values of the `day|night` group fall into category (− − −) making loop-closure detection not possible. We think that for signatures containing the kurtosis, this effect is due to disturbing images with Gaussian



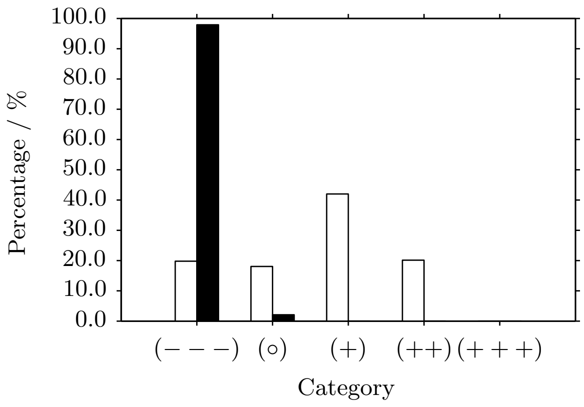
(1) `four-norm`



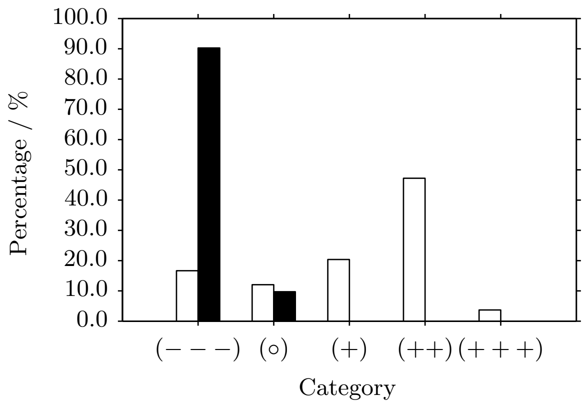
(2) `stat-norm`



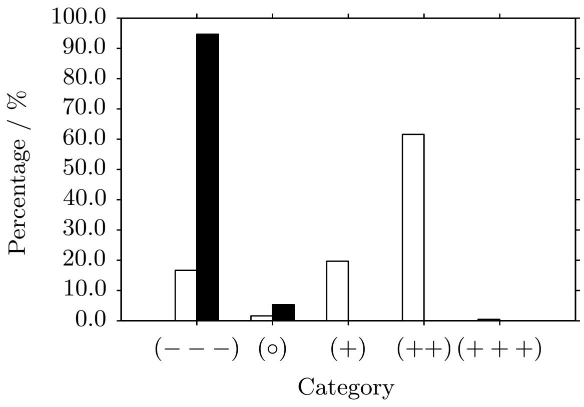
(3) `hist-norm`



(4) `hist-bin`



(5) `hist-cbin`



(6) `hist-tcbin`

Figure 6.6.: Frequency distributions of AUC values for signature-based loop-closure detection. Subfigures (1) to (6) depict the distributions for each of the six *evaluation* groups as defined in section 6.3.2. The distributions are used to identify promising methods for loop-closure detection as described in section 6.3.1.3. White and black bars depict the distributions for the `day|day` and `day|night` *test* groups, respectively. Binning was done w.r.t. the categories defined in table 5.3 as follows: (− − −) loop-closure detection not possible; (◦) moderate loop-closure detection performance; (+) good performance; (++) very good performance; (+ + +) ideal results (i.e. AUC = 1.0).

Table 6.3.: Best methods w.r.t. loop-closure detection for `four-norm`, i.e. for comparing Fourier signatures with norm functions. The table lists the the methods which achieve best possible performance for *both* test groups `day|day` and `day|night`. See section 6.3.1.3 for a description how these methods were identified based on figure 6.6.1. Categories for AUC values are defined in table 6.2 and given in the form $^{\texttt{day|day}}/_{\texttt{day|night}}$.

Category	s	d	r	b	$r \cdot b$	Test groups	
						`day|day`	`day|night`
						AUC	AUC
+++/○	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	32	8	256	1.000	0.612
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	32	16	512	1.000	0.612
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	32	32	1024	1.000	0.612
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	16	8	128	1.000	0.611
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	4	8	32	1.000	0.610
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	16	16	256	1.000	0.610
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	16	32	512	1.000	0.610
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	4	16	64	1.000	0.609
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	4	32	128	1.000	0.609
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	32	4	128	1.000	0.608
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	8	8	64	1.000	0.607
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	8	16	128	1.000	0.607
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	8	32	256	1.000	0.607
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	16	4	64	1.000	0.607
	$s_{\mathtt{afc}}$	$d_{\mathtt{eucl}}$	32	32	1024	1.000	0.606
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	4	4	16	1.000	0.606
	$s_{\mathtt{afc}}$	$d_{\mathtt{eucl}}$	16	32	512	1.000	0.605
	$s_{\mathtt{afc}}$	$d_{\mathtt{maxn}}$	8	4	32	1.000	0.605
	$s_{\mathtt{afc}}$	$d_{\mathtt{eucl}}$	32	16	512	1.000	0.604
	$s_{\mathtt{afc}}$	$d_{\mathtt{eucl}}$	8	32	256	1.000	0.603
	$s_{\mathtt{afc}}$	$d_{\mathtt{eucl}}$	16	16	256	1.000	0.603
	$s_{\mathtt{afc}}$	$d_{\mathtt{eucl}}$	8	16	128	1.000	0.601
	$s_{\mathtt{afc}}$	$d_{\mathtt{eucl}}$	32	8	256	1.000	0.600
	$s_{\mathtt{afc}}$	$d_{\mathtt{man}}$	16	32	512	1.000	0.600
	$s_{\mathtt{afc}}$	$d_{\mathtt{man}}$	32	32	1024	1.000	0.600

noise (section 5.3.1.1): As computing the kurtosis involves the computation of the forth power of the deviation from the average image brightness, the kurtosis is strongly influenced by low-frequency image content such as noise (ZORAN and WEISS [727]). For this reason, we do not further evaluate the performance of statistical signatures.

Among the group of methods comparing histogram-based signatures with norm functions (`hist-norm`; figure 6.6.3), approximately 50 % of the AUC values for the `day|day` group fall into category (++), but none of the methods achieves perfect classification (+ + +). The best AUC values for the `day|night` group were binned into category (○). Thus, the best possible combination belongs to categories (++) and (○) for the `day|day` and the `day|night` groups, respectively. A total of 12 methods are identified (table 6.4) with AUC values between 0.600 and 0.612. All combinations rely on $b = 4$ histogram bins with each of the signatures $s_{\mathtt{hist}}$ and $s_{\mathtt{chist}}$ accounting for 50 % of the cases. As dissimilarity functions, 2 of 12 methods compare signatures by the Manhattan norm $d_{\mathtt{man}}$, 5 by the Euclidean distance $d_{\mathtt{eucl}}$ and 5 by the maximum norm $d_{\mathtt{maxn}}$. With an AUC values of 0.997 and 0.612 for the `day|day`- and the `day|night` group, the best performance is achieved by combining the cumulative histogram signature $s_{\mathtt{chist}}$, the maximum norm $d_{\mathtt{maxn}}$, $r = 32$ rings and $b = 4$ histogram bins.

For comparing histogram-based signatures with bin-by-bin matching functions (`hist-bin`; figure 6.6.4), the best methods for the `day|day` group belong to category (++); none of the methods achieves an ideal classification result. The best method for the `day|night` group again belong to category (○). Table 6.5 lists 6 different methods belong to categories (++) and (○) for the

Table 6.4.: Best methods w.r.t. loop-closure detection for `hist-norm`, i.e. for comparing histogram-based signatures with norm functions. The table lists the the methods which achieve best possible performance for *both* test groups `day|day` and `day|night`. See section 6.3.1.3 for a description how these methods were identified based on figure 6.6.3. Categories for AUC values are defined in table 6.2 and given in the form $^{\texttt{day|day}}/_{\texttt{day|night}}$.

Category						Test groups	
						`day|day`	`day|night`
	s	d	r	b	$r \cdot b$	AUC	AUC
++/∘	$\mathrm{s}_{\mathtt{chist}}$	$\mathrm{d}_{\mathtt{maxn}}$	32	4	128	0.997	0.612
	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{eucl}}$	32	4	128	0.997	0.609
	$\mathrm{s}_{\mathtt{chist}}$	$\mathrm{d}_{\mathtt{eucl}}$	32	4	128	0.997	0.607
	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{man}}$	32	4	128	0.997	0.603
	$\mathrm{s}_{\mathtt{chist}}$	$\mathrm{d}_{\mathtt{man}}$	32	4	128	0.997	0.601
	$\mathrm{s}_{\mathtt{chist}}$	$\mathrm{d}_{\mathtt{maxn}}$	16	4	64	0.996	0.608
	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{eucl}}$	16	4	64	0.996	0.605
	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{maxn}}$	32	4	128	0.996	0.604
	$\mathrm{s}_{\mathtt{chist}}$	$\mathrm{d}_{\mathtt{eucl}}$	16	4	64	0.996	0.603
	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{maxn}}$	16	4	64	0.995	0.600
	$\mathrm{s}_{\mathtt{chist}}$	$\mathrm{d}_{\mathtt{maxn}}$	8	4	32	0.994	0.606
	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{eucl}}$	8	4	32	0.994	0.603

Table 6.5.: Best methods w.r.t. loop-closure detection for `hist-bin`, i.e. for comparing histogram-based signatures with bin-by-bin matching functions. The table lists the the methods which achieve best possible performance for *both* test groups `day|day` and `day|night`. See section 6.3.1.3 for a description how these methods were identified based on figure 6.6.4. Categories for AUC values are defined in table 6.2 and given in the form $^{\texttt{day|day}}/_{\texttt{day|night}}$.

Category						Test groups	
						`day|day`	`day|night`
	s	d	r	b	$r \cdot b$	AUC	AUC
++/∘	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{chi}}$	32	4	128	0.996	0.608
	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{kl}}$	32	4	128	0.996	0.604
	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{chi}}$	16	4	64	0.995	0.606
	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{kl}}$	16	4	64	0.995	0.603
	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{chi}}$	8	4	32	0.993	0.605
	$\mathrm{s}_{\mathtt{hist}}$	$\mathrm{d}_{\mathtt{chi}}$	4	4	16	0.990	0.603

`day|day` and the `day|night` group, respectively. All combinations rely on $b = 4$ histogram bins and achieve AUC values between 0.603 and 0.608. 2 of the 6 combinations compare signatures by the Kullback-Leibler divergence $\mathrm{d}_{\mathtt{kl}}$, the remaining 4 combinations use the χ^2 dissimilarity function. The best performance of this group is obtained by the combination of $\mathrm{s}_{\mathtt{hist}}$, $\mathrm{d}_{\mathtt{chi}}$, $r = 32$, and $b = 4$ and yields AUC values of 0.996 and 0.608 for the `day|day` and `day|night` group, respectively.

Figure 6.6.5 visualizes the frequency distributions for comparing histogram-based signatures with cross-bin matching functions without thresholded ground distance (`hist-cbin`). For this group, only a small number of methods achieves perfect classification (+ + +) of the `day|day` group, most methods allow for very good classification (++). For the `day|night` group, all AUC values fall into categories (− − −) and (∘). Hence, the best possible methods of this group belong to categories (+ + +) and (∘) for the `day|day` and the `day|night` group, respectively. Only one method achieves AUC values belonging to this class (table 6.6): The method relying on histogram signatures $\mathrm{s}_{\mathtt{hist}}$, the quadratic form dissimilarity function $\mathrm{d}_{\mathtt{qf}}$, and $r = b = 32$. With an AUC value of 0.601 for the `day|night` group, its performance is at the lower limit of the category (∘). Further methods with similar loop-closure detection performance achieve a very good classification (++) of the `day|day` group and moderate performance of the `day|night` group. These methods all rely on the $\mathrm{s}_{\mathtt{hist}}$

Table 6.6.: Best methods w.r.t. loop-closure detection for `hist-cbin`, i.e. for comparing histogram-based signatures with cross-bin matching functions without thresholded ground distance. The table lists the the methods which achieve best possible performance for *both* test groups `day|day` and `day|night`. See section 6.3.1.3 for a description how these methods were identified based on figure 6.6.5. Categories for AUC values are defined in table 6.2 and given in the form $^{\texttt{day|day}}/_{\texttt{day|night}}$.

Category	s	d	r	b	$r \cdot b$	Test groups	
						`day|day`	`day|night`
						AUC	AUC
[1]	$s_{\mathtt{hist}}$	$d_{\mathtt{qfc}}$	32	32	1024	1.000	0.601
	$s_{\mathtt{hist}}$	$d_{\mathtt{qf}}$	32	16	512	0.999	0.607
	$s_{\mathtt{hist}}$	$d_{\mathtt{qf}}$	32	32	1024	0.999	0.607
	$s_{\mathtt{hist}}$	$d_{\mathtt{qf}}$	16	16	256	0.999	0.605
	$s_{\mathtt{hist}}$	$d_{\mathtt{qf}}$	16	32	512	0.999	0.605
	$s_{\mathtt{hist}}$	$d_{\mathtt{emd}}$	32	32	1024	0.999	0.604
	$s_{\mathtt{hist}}$	$d_{\mathtt{qf}}$	8	32	256	0.999	0.602
	$s_{\mathtt{hist}}$	$d_{\mathtt{qfc}}$	32	16	512	0.999	0.602
	$s_{\mathtt{hist}}$	$d_{\mathtt{qfc}}$	16	16	256	0.999	0.600
	$s_{\mathtt{hist}}$	$d_{\mathtt{qf}}$	32	4	128	0.998	0.606
++/○	$s_{\mathtt{hist}}$	$d_{\mathtt{qf}}$	32	8	256	0.998	0.604
	$s_{\mathtt{hist}}$	$d_{\mathtt{emd}}$	16	32	512	0.998	0.602
	$s_{\mathtt{hist}}$	$d_{\mathtt{qf}}$	8	16	128	0.998	0.602
	$s_{\mathtt{hist}}$	$d_{\mathtt{qf}}$	16	8	128	0.998	0.602
	$s_{\mathtt{hist}}$	$d_{\mathtt{qfc}}$	32	4	128	0.998	0.601
	$s_{\mathtt{hist}}$	$d_{\mathtt{qfc}}$	16	4	64	0.998	0.600
	$s_{\mathtt{hist}}$	$d_{\mathtt{qf}}$	16	4	64	0.997	0.604
	$s_{\mathtt{hist}}$	$d_{\mathtt{emd}}$	32	16	512	0.997	0.603
	$s_{\mathtt{hist}}$	$d_{\mathtt{qf}}$	8	4	32	0.996	0.602
	$s_{\mathtt{hist}}$	$d_{\mathtt{emd}}$	32	8	256	0.994	0.603
	$s_{\mathtt{hist}}$	$d_{\mathtt{emd}}$	8	16	128	0.989	0.600

[1] + + +/○

signature and yield AUC values between 0.600 and 0.607. Among the 20 methods, images are compared by $d_{\mathtt{qfc}}$ in 4 cases, by $d_{\mathtt{emd}}$ in 5 cases and by $d_{\mathtt{qf}}$ in 11 cases. The best performance with AUC values of 0.999 and 0.607 for the `day|day` and the `day|night` group is obtained by the $s_{\mathtt{hist}}$ signature with $r = 32$ rings, $b \in \{16, 32\}$ histogram bins and comparing signatures by the quadratic form dissimilarity function $d_{\mathtt{qf}}$.

The last of the six groups, `hist-tcbin`, subsumes the methods comparing histogram-based signatures by cross-bin matching functions with thresholded ground distance (figure 6.6.6). For the `day|day` group, 0.5 % and approximately 60 % of the AUC values fall into categories (+ + +) and (++), thus achieving perfect or nearly perfect classification. Most of the AUC values obtained for the `day|night` group belong to category (− − −), only approximately 4 % belong to category (○). Theoretically, the best possible combinations of AUC categories category (+ + +) for the `day|day` group and category (○) for the `day|night` test group. However, none of the AUC values fall into this category. This case could have been better identified by a 2D histogram of the joint distribution of AUC values for the `day|day` and `day|night` groups. The most promising methods of this class therefore belong to category (+ + +) for the `day|day` group and to category (○) for the `day|night` test group. The 20 methods of this combination (table 6.7) all rely on the histogram signature $s_{\mathtt{hist}}$ and obtain AUC values ranging from 0.993 and 0.999 and from 0.600 to 0.609 for the `day|day` and `day|night` test groups, respectively. For comparing signatures, 3 of the 23 combinations rely on the thresholded earth mover's distance $d_{\mathtt{emdt*}}$, 8 on the thresholded quadratic χ^2 function $d_{\mathtt{qfct*}}$ and the remaining 12 on the thresholded quadratic-form function $d_{\mathtt{qft*}}$. In case the best combination is determined depending on the performance for the `day|day` group, the combination of $s_{\mathtt{hist}}$, $d_{\mathtt{qfct4}}$,

Table 6.7.: Best methods w.r.t. loop-closure detection for `hist-tcbin`, i.e. for comparing histogram-based signatures with cross-bin matching functions with thresholded ground distance. The table lists the the methods which achieve best possible performance for *both* test groups `day|day` and `day|night`. See section 6.3.1.3 for a description how these methods were identified based on figure 6.6.6. Categories for AUC values are defined in table 6.2 and given in the form $^{\texttt{day|day}}/_{\texttt{day|night}}$.

Category	s	d	r	b	$r \cdot b$	Test groups `day|day` AUC	Test groups `day|night` AUC
++/○	$s_{\texttt{hist}}$	$d_{\texttt{qfct4}}$	32	16	512	0.999	0.607
	$s_{\texttt{hist}}$	$d_{\texttt{emdt4}}$	32	8	256	0.999	0.604
	$s_{\texttt{hist}}$	$d_{\texttt{emdt4}}$	32	16	512	0.999	0.604
	$s_{\texttt{hist}}$	$d_{\texttt{qfct4}}$	16	16	256	0.999	0.604
	$s_{\texttt{hist}}$	$d_{\texttt{qft2}}$	32	4	128	0.998	0.609
	$s_{\texttt{hist}}$	$d_{\texttt{qft4}}$	32	8	256	0.998	0.608
	$s_{\texttt{hist}}$	$d_{\texttt{qft4}}$	32	4	128	0.998	0.606
	$s_{\texttt{hist}}$	$d_{\texttt{qfct2}}$	32	8	256	0.998	0.604
	$s_{\texttt{hist}}$	$d_{\texttt{qft4}}$	32	16	512	0.998	0.604
	$s_{\texttt{hist}}$	$d_{\texttt{qfct4}}$	8	16	128	0.998	0.602
	$s_{\texttt{hist}}$	$d_{\texttt{emdt4}}$	16	8	128	0.998	0.601
	$s_{\texttt{hist}}$	$d_{\texttt{qfct4}}$	32	4	128	0.998	0.601
	$s_{\texttt{hist}}$	$d_{\texttt{qfct4}}$	16	4	64	0.998	0.600
	$s_{\texttt{hist}}$	$d_{\texttt{qfct4}}$	32	8	256	0.998	0.600
	$s_{\texttt{hist}}$	$d_{\texttt{qft4}}$	16	16	256	0.998	0.600
	$s_{\texttt{hist}}$	$d_{\texttt{qft2}}$	16	4	64	0.997	0.607
	$s_{\texttt{hist}}$	$d_{\texttt{qft4}}$	16	8	128	0.997	0.606
	$s_{\texttt{hist}}$	$d_{\texttt{qft4}}$	16	4	64	0.997	0.604
	$s_{\texttt{hist}}$	$d_{\texttt{qfct2}}$	16	8	128	0.997	0.602
	$s_{\texttt{hist}}$	$d_{\texttt{qft2}}$	8	4	32	0.996	0.604
	$s_{\texttt{hist}}$	$d_{\texttt{qft4}}$	8	4	32	0.996	0.602
	$s_{\texttt{hist}}$	$d_{\texttt{qft4}}$	8	8	64	0.996	0.602
	$s_{\texttt{hist}}$	$d_{\texttt{qft2}}$	4	4	16	0.993	0.600

$r = 32$ and $b = 16$ performs best with AUC values of 0.999 and 0.607 for the `day|day` and the `day|night` group, respectively. With AUC values of 0.998 and 0.609, the combination of $s_{\texttt{hist}}$, $d_{\texttt{qft2}}$, $r = 32$ and $b = 4$ performs best if combinations are determined based on the AUC values obtained for the `day|night` group.

From the results summarized in figure 6.6, we conclude that Fourier-based signatures and histogram-based signatures in conjunction with a comparison by a standard norm function yield the best performance. As comparing histogram-based signatures by histogram-comparison functions does not achieve a better performance than comparing histogram-based signatures by norm functions, we do not further evaluate histogram-specific dissimilarity functions. Among the best combinations of comparing Fourier-based and histogram-based signatures by norm functions, the best AUC values were often obtained by the maximum norm $d_{\texttt{maxn}}$. We therefore limit the further evaluation to comparing Fourier-based and histogram-based signatures by the maximum norm. Because none of the tested combinations achieved AUC values above 0.6 for the `day|night` group, we do not consider statistical signatures for the following evaluation step.

6.3.2.2. Performance of Loop-Closure Detection Depending on the Signature Dimensionality

As a second and more detailed evaluation step, we analyze for the most promising methods tested in this chapter how the AUC values depend on the signature's dimensionality. The results of this evaluation are shown in figures 6.7 and 6.8 for histogram-based and Fourier-based signatures, respectively. The charts visualize the AUC values vs. the signature's overall dimensionality $r \cdot b$. As

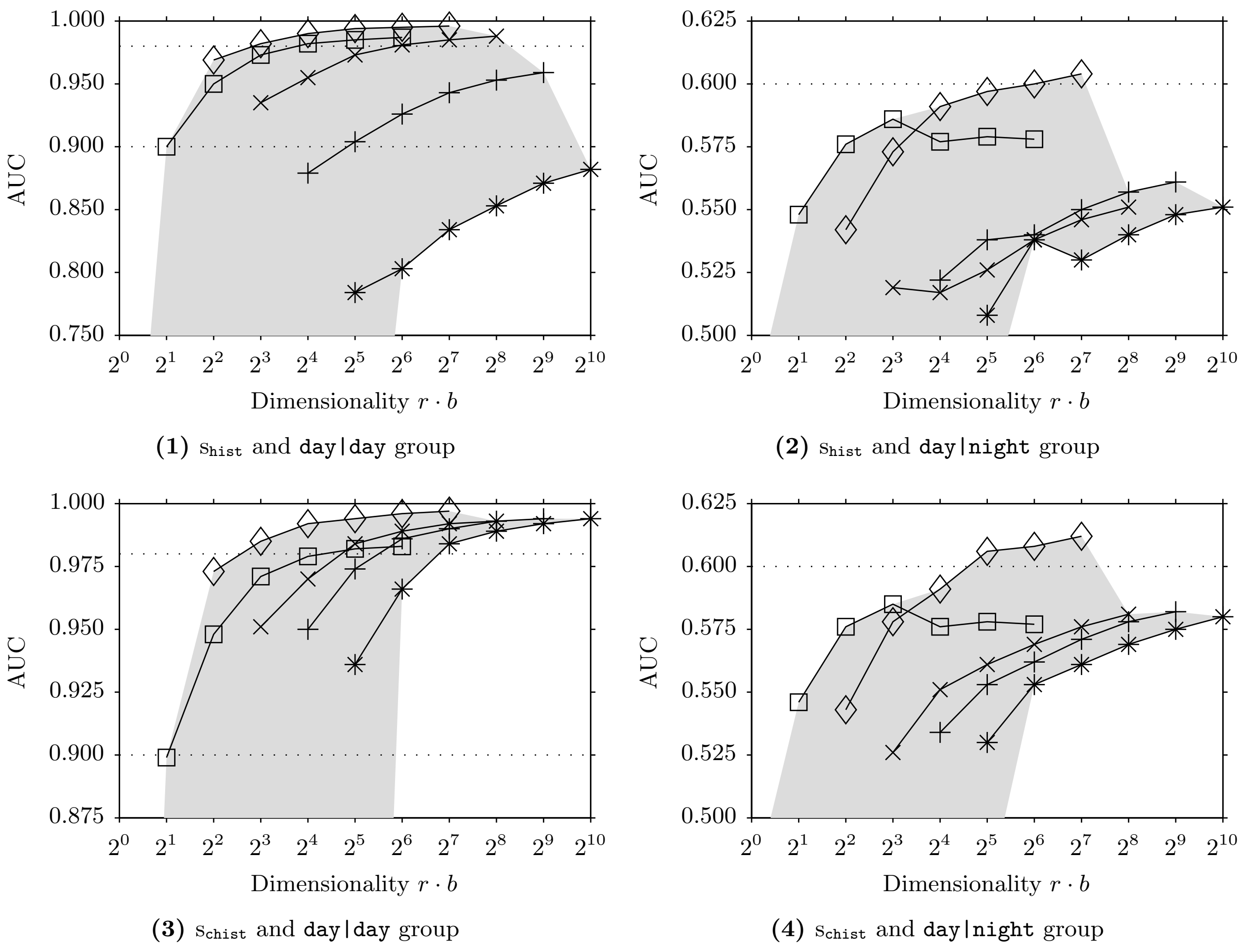
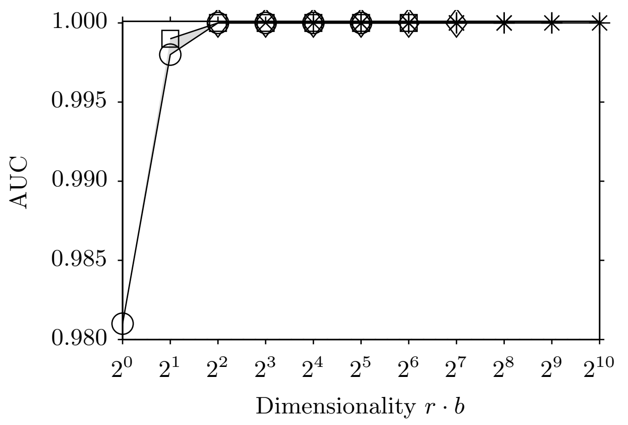


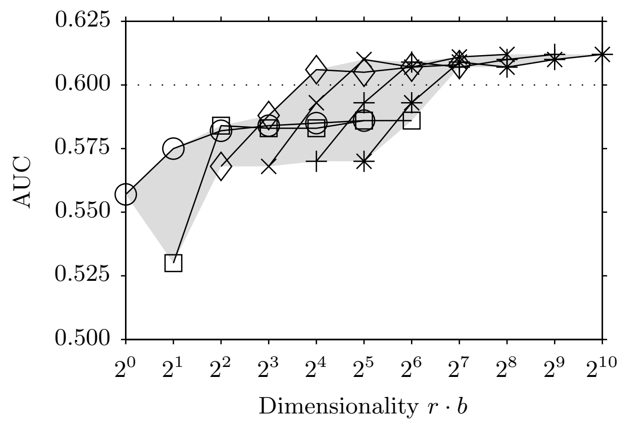
Figure 6.7.: AUC values depending on the signatures' dimensionality for histogram-based signatures $s_{\texttt{afc}}$ and $s_{\texttt{zafc}}$ compared by the maximum norm $d_{\texttt{maxn}}$. In the top row (subfigures (1) and (2)), the results for the intensity histogram signature $s_{\texttt{hist}}$ are shown; the bottom row (subfigures (3) and (4)) depicts the results for the comulative histogram signature $s_{\texttt{chist}}$. The left column (subfigures (1) and (3)) contains the results for the `day|day` test group, the right column (subfigures (2) and (4)) visualize the results for the `day|night` test group. The x-axis of each plot represents the signature's overall dimensionality (i.e. $r \cdot b$) in a logarithmic scale with base 2; the y-axis carries the AUC values. The range between the maximum (best) and minimum (worst) AUC values is depicted by the shaded area. The number of bins b is coded by the marker: ($\circ$) $b = 1$ (not visible), ($\square$) $b = 2$, ($\diamond$) $b = 4$, ($\times$) $b = 8$, ($+$) $b = 16$, and ($*$) $b = 32$. For histograms with a single bin, loop-closure detection is not possible because all intensities are grouped into the same bin. Thus, the results are at chance level (0.5) or, as in the depicted cases, slightly below (due small numerical inaccuracies). They are therefore not visible in the figures, but contribute to the shaded area. The horizontal dotted lines mark the limits of the performance categories as defined by table 6.2.

several combinations of r and b can result in identical overall dimensionalities $r \cdot b$, the number b of histogram bins or Fourier coefficients is coded by different markers as explained in the captions of figures 6.7 and 6.8. The shaded area marks for each dimensionality the range between the maximum (best) and minimum (worst) AUC values.

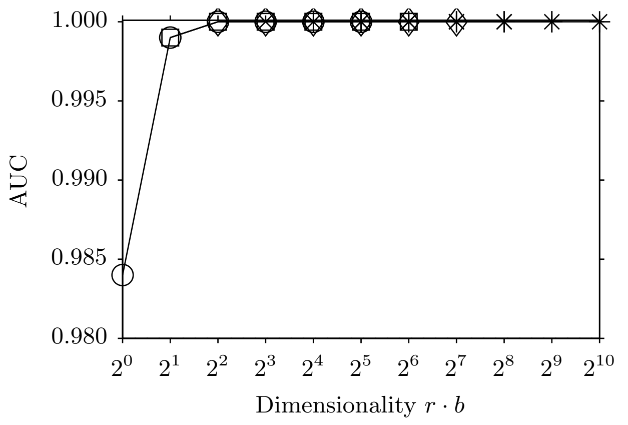
The results of the histogram-based signatures $s_{\texttt{hist}}$ and $s_{\texttt{chist}}$ are depicted in figure 6.7. In comparison to the `day|day` group, the performance of the signatures obtained for the `day|night` group is strongly decreased as already observed in section 6.3.2.1. For both signatures and both test groups, the best AUC value is obtained for a dimensionality of $2^7 = 128$ dimensions. In all cases, the best performance is obtained for $b = 4$ ($\diamond$) histogram bins. Thus, to achieve an overall dimensionality of 128 dimensions, $r = 32$ rings were used. It seems that the methods with a small number of histogram bins b and a larger number of rings r achieve the best results because for



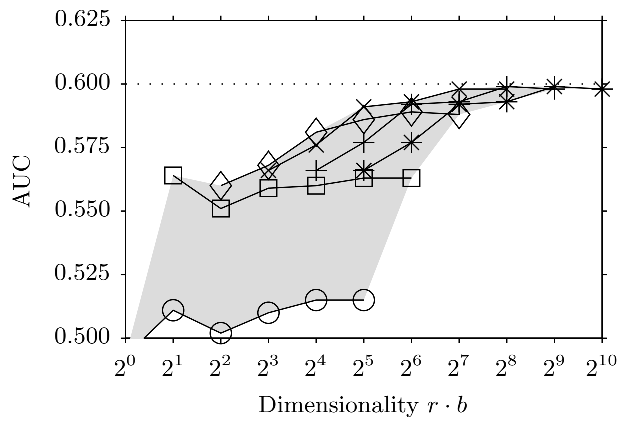
(1) $s_{\mathtt{afc}}$ and `day|day` group



(2) $s_{\mathtt{afc}}$ and `day|night` group



(3) $s_{\mathtt{zafc}}$ and `day|day` group



(4) $s_{\mathtt{zafc}}$ and `day|night` group

Figure 6.8.: AUC values depending on the signatures' dimensionality for Fourier-based signatures $s_{\mathtt{afc}}$ (top row; subfigures (1) and (2)) and $s_{\mathtt{zafc}}$ (bottom row; subfigures (3) and (4)) compared by the maximum norm $d_{\mathtt{maxn}}$. The left (subfigures (1) and (3)) and right (subfigures (2) and (4)) columns depict the results for the `day|day` and `day|night` test groups, respectively. The x-axis of each plot represents the signature's overall dimensionality (i.e. $r \cdot b$) in a logarithmic scale with base 2; the y-axis carries the AUC values. The range between the maximum (best) and minimum (worst) AUC values is depicted by the shaded area. The number of Fourier coefficients b is coded by the marker: ($\circ$) $b = 1$, ($\square$) $b = 2$, ($\diamond$) $b = 4$, ($\times$) $b = 8$, ($+$) $b = 16$, and ($\ast$) $b = 32$. The horizontal dotted lines mark the limits of the performance categories as defined by table 6.2.

increasing the dimensionality, the performance decreases again. For histograms with only one bin ($b = 1$, $\circ$) loop-closure detection is not possible because all pixel intensities are grouped into the same bin. The resulting AUC values are around chance level or below 0.5 and are truncated in figure 6.7. For the cases allowing for loop-closure detection, the worst performance is for both test groups and both signatures obtained for $b = 32$ ($\ast$) histogram bins. The performance for $b = 8$ ($\times$) and $b = 16$ ($+$) histogram bins is between the performance of the better parameter choices and the performance of $b = 32$. The performance differences between the better parameter choices ($b = 2$ or $b = 4$) and the worst choice ($b = 32$) are stronger for the signature $s_{\mathtt{hist}}$ based on gray-value histograms (figures 6.7.1 and 6.7.2) than for the signature $s_{\mathtt{chist}}$ based on cumulative histograms (figures 6.7.3 and 6.7.4).

Figure 6.8 shows the results obtained for the Fourier-based signatures $s_{\mathtt{afc}}$ (figures 6.8.1 and 6.8.3) and $s_{\mathtt{zafc}}$ (figures 6.8.2 and 6.8.4), respectively. For the `day|day` group and more than 4 dimensions, the signatures $s_{\mathtt{afc}}$ and $s_{\mathtt{zafc}}$ obtain perfect classification (AUC=1.0) independent of the specific combination of r and b. For less than 4 dimensions, both methods still obtain AUC values above 0.98.

Compared to the day|day group, the performance of the day|night group is strongly decreased. For the day|night group and the $s_{\texttt{hist}}$ signature (figure 6.8.2), the best AUC values are approximately 0.61 and were obtained for 4 and more dimensions. AUC values close to 0.61 were obtained by several combinations of rings r and Fourier coefficients b; the maximum AUC value is 0.612, but it is not pronounced. It was obtained for $r = 32$ rings in combination with $b \in \{8, 16, 32\}$ Fourier coefficients resulting in overall dimensionalities of 256, 512 and 1024. Thus, the $s_{\texttt{afc}}$ signature seems to perform best for a larger number of rings r and Fourier coefficients b. The performance range for the $s_{\texttt{zafc}}$ signature and the day|night group (figure 6.8.4) is larger than for the $s_{\texttt{afc}}$ signature. For 64 and more dimensions, the best AUC values are close together. The maximum AUC value of 0.599 is not pronounced and is obtained for $r = 16$ in conjunction with $b \in \{16, 32\}$. The $s_{\texttt{zafc}}$ signature is not as robust against changes of the illumination as it was expected from the theoretical results of section 6.2.4. It even performs slightly worse than the signature $s_{\texttt{afc}}$ including the first Fourier coefficient.

From these results we conclude that —at least for the day|day test group— reliable loop-closure detection is possible with signatures of more than 32 dimensions. As already mentioned in section 6.3.2.1, the performance of loop-closure detection is strongly decreased for the day|night group containing strong changes of the illumination. The best performance is obtained for signatures with an overall dimensionality of $2^7 = 128$ and of $2^8 = 256$ dimensions for histogram-based and Fourier-based signatures, respectively. It seems that the histogram-based signatures achieve the best performance for a large number of rings r and a small number of histogram bins b whereas the best performance of the Fourier-based signatures was obtained for large r and large b. The specific choice of the parameters r and b has a smaller influence on the resulting performance for Fourier-based signatures than for histogram-based signatures. Among the Fourier-based signatures, the signature $s_{\texttt{afc}}$ containing the first Fourier coefficient depends less on the choice of r and b than the $s_{\texttt{zafc}}$ signature, which does not contain the first coefficient.

6.3.2.3. Robustness Against Perceptual Aliasing and Perceptual Variability

In order to assess the methods' robustness against perceptual aliasing and perceptual variability, the data was evaluated as outlined in section 6.3.1.3. Computing the performance measures $P_{\texttt{corr}}$, $P_{\texttt{misc}}$, $P_{\texttt{misf}}$, dist_{avg}, and dist_{med} allows us to draw conclusions how often mismatches occur and how close they are to the expected match.

Among the 1548 tested combinations, 93 achieve perfect results for the day|day dataset, i.e. for image pairs acquired under nearly constant illumination conditions; a total of 311 combinations achieve a percentage of correct classification $P_{\texttt{corr}}$ of greater than or equal to 95 %. In contrast to that, the best combination of the day|night test group (containing image pairs simulating strong illumination changes) only achieves correct classifications in 3.5 % of the cases. This strong performance difference is again due to the methods' low tolerance against changes of the illumination we already observed when evaluating the performance of loop-closure detection (sections 6.3.2.1 and 6.3.2.2).

To get a more detailed impression of the results, we categorize the results into six *evaluation* groups as defined in section 6.3.2: four-norm, stat-norm, hist-norm, hist-bin, hist-cbin, and hist-tcbin. For each of the six groups and for each of the two test groups, we identified the best method based on the percentage of correct detections $P_{\texttt{corr}}$. If two or more methods obtain identical values of $P_{\texttt{corr}}$ (including ideal results of $P_{\texttt{corr}} = 100\,\%$) for a test group, the $P_{\texttt{corr}}$ value of the other group is used for comparison. The best methods for each test group are summarized in tables 6.8.1 and 6.8.2 for the day|day and the day|night group, respectively. The tables also contain the performance measures of the other test group, and a rating whether the method is applicable for loop-closure detection. We consider a method to be applicable if the conditions $P_{\texttt{corr}} > 90\,\%$ and $\text{dist}_{\text{med}} < 15\,\text{cm}$ hold. The maximum distance of 15 cm was chosen because it is half the inter-lane

Table 6.8.: Best methods w.r.t. robustness against perceptual aliasing and perceptual variability. The subtables (1) and (2) contain for each of the six *evaluation* groups (`four-norm`, `stat-norm`, `hist-norm`, `hist-bin`, `hist-cbin`, and `hist-tcbin`; section 6.3.2)) the combination achieving the largest percentage of correct loop-closure detections $P_{\texttt{corr}}$ for the `day|day` and `day|night` test group, respectively. In case several methods of a class obtained identical values of $P_{\texttt{corr}}$ for a test group, the combination achieving the best performance (largest AUC value) for the other test group was used. The value used for identifying the best methods of each group is highlighted in the corresponding subtable. Please note that the columns in the subtables are not ranked w.r.t. $P_{\texttt{corr}}$, but that the evaluation groups are given in the order as they were defined in section 6.3.2. We consider a method to be applicable (✓) for loop-closure detection of an autonomous cleaning robot if $P_{\texttt{corr}} > 90\,\%$ and $\text{dist}_{\text{med}} < 15\,\text{cm}$; methods which we consider to be not applicable are marked by a ✗.

(1) `day|day` group.

		Evaluation groups					
		`four-norm`	`stat-norm`	`hist-norm`	`hist-bin`	`hist-cbin`	`hist-tcbin`
	s	$\texttt{s}_{\texttt{afc}}$	$\texttt{s}_{\texttt{mmv}}$[1]	$\texttt{s}_{\texttt{chist}}$	$\texttt{s}_{\texttt{hist}}$	$\texttt{s}_{\texttt{hist}}$	$\texttt{s}_{\texttt{chist}}$
	d	$\texttt{d}_{\texttt{man}}$	$\texttt{d}_{\texttt{maxn}}$	$\texttt{d}_{\texttt{man}}$	$\texttt{d}_{\texttt{bhat}}$	$\texttt{d}_{\texttt{qfc}}$	$\texttt{d}_{\texttt{emdt2}}$
	r	32	32	32	32	32	32
	b	32	2	32	4	32	32
	$r \cdot b$	1024	64	1024	128	1024	1024
`day|day`	$P_{\texttt{corr}}$ / %	100.0	100.0	99.9	88.5	99.6	99.8
	$P_{\texttt{misc}}$ / %	0.0	0.0	0.1	9.7	0.4	0.2
	$P_{\texttt{misf}}$ / %	0.0	0.0	0.0	1.8	0.0	0.0
	dist_{avg} / cm	0.0	0.0	10.0	12.4	10.7	11.4
	dist_{med} / cm	0.0	0.0	10.0	10.0	10.0	10.0
	Appplicability	✓	✓	✓	✗	✓	✓
`day|night`	$P_{\texttt{corr}}$ / %	2.6	2.0	1.8	1.2	1.4	2.0
	$P_{\texttt{misc}}$ / %	9.3	9.7	11.3	8.0	12.0	11.3
	$P_{\texttt{misf}}$ / %	88.1	88.3	87.0	90.8	86.6	86.7
	dist_{avg} / cm	66.4	61.2	56.7	62.0	56.0	57.1
	dist_{med} / cm	62.0	53.9	50.0	51.0	50.5	50.0
	Appplicability	✗	✗	✗	✗	✗	✗

[1] For statistical signatures, b is the subdescriptor dimensionality (section 6.2.3).

(2) `day|night` group.

		Evaluation groups					
		`four-norm`	`stat-norm`	`hist-norm`	`hist-bin`	`hist-cbin`	`hist-tcbin`
	s	$\texttt{s}_{\texttt{afc}}$	$\texttt{s}_{\texttt{mmv}}$[1]	$\texttt{s}_{\texttt{chist}}$	$\texttt{s}_{\texttt{hist}}$	$\texttt{s}_{\texttt{chist}}$	$\texttt{s}_{\texttt{chist}}$
	d	$\texttt{d}_{\texttt{man}}$	$\texttt{d}_{\texttt{maxn}}$	$\texttt{d}_{\texttt{maxn}}$	$\texttt{d}_{\texttt{chi}}$	$\texttt{d}_{\texttt{qf}}$	$\texttt{d}_{\texttt{emdt4}}$
	r	32	16	32	32	16	16
	b	32	2	4	32	32	8
	$r \cdot b$	1024	32	128	1024	512	128
`day|day`	$P_{\texttt{corr}}$ / %	100.0	99.7	83.2	29.7	97.9	88.9
	$P_{\texttt{misc}}$ / %	0.0	0.2	13.5	47.2	2.1	9.6
	$P_{\texttt{misf}}$ / %	0.0	0.1	3.2	23.1	0.0	1.5
	dist_{avg} / cm	0.0	12.0	13.4	17.2	10.4	12.4
	dist_{med} / cm	0.0	10.0	10.0	10.0	10.0	10.0
	Appplicability	✓	✓	✗	✗	✓	✗
`day|night`	$P_{\texttt{corr}}$ / %	2.6	2.1	2.5	2.5	3.5	2.7
	$P_{\texttt{misc}}$ / %	9.3	10.1	10.2	10.8	14.6	12.7
	$P_{\texttt{misf}}$ / %	88.1	87.8	87.3	86.7	81.9	84.6
	dist_{avg} / cm	66.4	61.4	59.6	58.8	50.8	53.7
	dist_{med} / cm	62.0	53.9	50.0	51.0	41.2	44.7
	Appplicability	✗	✗	✗	✗	✗	✗

[1] For statistical signatures, b is the subdescriptor dimensionality (section 6.2.3).

distance (section 4.4). We expect that a smaller percentage of correct detections or larger distances between mismatches would strongly influence the navigation capabilities and thus consider them to be not applicable for reliable loop-closure detection.

Regarding the best methods for the `day|day` group, (table 6.8.1), five of the six combinations achieve perfect (`four-norm` and `stat-norm`) or almost perfect (`hist-norm`, `hist-cbin`, and `hist-tcbin`) results for the `day|day` group. If mismatches occur for these five methods, they are spatially very close to the expected match. Only the best method for `hist-bin`, i.e. comparing histogram-based signatures in combination with a bin-by-bin comparison function, performs considerably worse. For this combination, namely comparing the $\mathrm{s}_{\mathtt{hist}}$ signature with the Bhattacharyya coefficient $\mathrm{d}_{\mathtt{bhat}}$, mismatches occur in 11.5 % of the cases. Although most of the mismatches are "close" mismatches and although $\mathrm{dist}_{\mathrm{med}}$ and $\mathrm{dist}_{\mathrm{avg}}$ are relatively small, we consider the method to be not applicable for loop-closure detection of an autonomous floor-cleaning robot. In contrast to their performance for the `day|day` group, the performance for the `day|night` is much worse (table 6.8.1, lower part). None of the methods is applicable for reliable loop-closure detection: loop-closures are only detected correctly in approximately 2 % of the cases, and 85 % and more are distant mismatches. The spatial distance between expected match and computed match is in all cases larger than 50 cm. Beyond that, it is surprising that all methods rely on relatively large signatures.

None of the best methods identified for the `day|night` group (table 6.8.2) are applicable to loop-closure detection. These methods only correctly detect loop-closures for at most 3.5 % of the image pairs contained in the `day|night` group. More than 80 % of the mismatches are "far" mismatches, and the distance between expected match and computed match is in all cases larger than 40 cm. The performance of these methods for the `day|night` group is considerably better than for the `day|night` group, but the methods do not achieve the performance of the best methods identified for the `day|night` group (table 6.8.1). Three of six combinations achieve perfect (`four-norm`) or close to perfect (`stat-norm` and `hist-cbin`) results. The remaining three methods (`hist-tcbin`, `hist-norm`, and `hist-bin`) fail our criterion for applicability because the percentage of correct matches is too small. Nevertheless, the mean and median spatial distances between expected match and computed match are in three cases surprisingly small.

This overview of the results clearly reveals that (i) several combinations achieve very good or perfect results for the `day|day` dataset, but that (ii) none of the tested methods can cope with the drastic illumination changes contained in the `day|night` group. Thus, not even the Fourier-based signature omitting the DC component ($sig_{\mathtt{zafc}}$) and histogram-based signatures in conjunction with cross-bin matching functions ($\mathrm{d}_{\mathtt{qf}}$, $\mathrm{d}_{\mathtt{qfc}}$, $\mathrm{d}_{\mathtt{emd}}$, $\mathrm{d}_{\mathtt{qft}}$, $\mathrm{d}_{\mathtt{qfct}}$, and $\mathrm{d}_{\mathtt{emdt}}$) can compensate for the illumination changes of the `day|night` group. The best methods summarized in table 6.8 were identified based on their $P_{\mathtt{corr}}$ values for *one* of the test groups. Since not even the best methods of the `day|night` group are considered to be applicable on a real cleaning robot, there cannot exist a combination of signature function s and dissimilarity function d, which fulfills the criterion for *both* test groups. With these results, we consider a more detailed evaluation, e.g. investigating how the signature dimensionality influences the robustness against perceptual aliasing and perceptual variability, to be not reasonable. The results rather suggest to increase the methods' robustness against changes of the illumination or —as we expect such drastic changes to be unlikely during a single cleaning run of the robot— to test the methods under more realistic illumination conditions. For the latter reason, real-robot experiments will be performed (section 6.4).

6.3.2.4. Computational Aspects of Signature Functions

In this section we analyze the computing time required to derive a single signature from a panoramic image. Computing times were determined for each signature function s and possible parameter combinations as defined in section 6.3.1.3. All results presented in the following are median

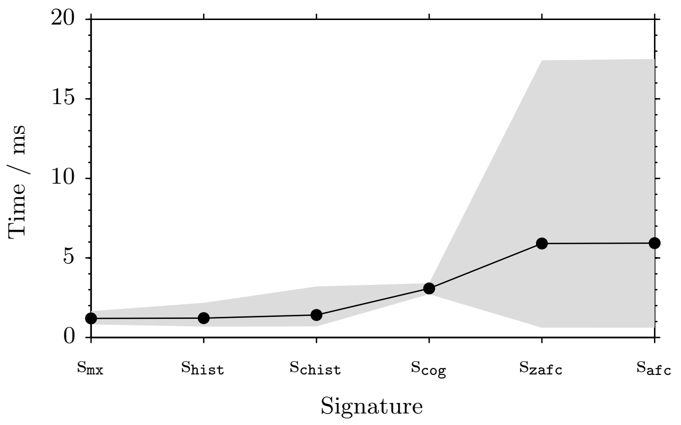


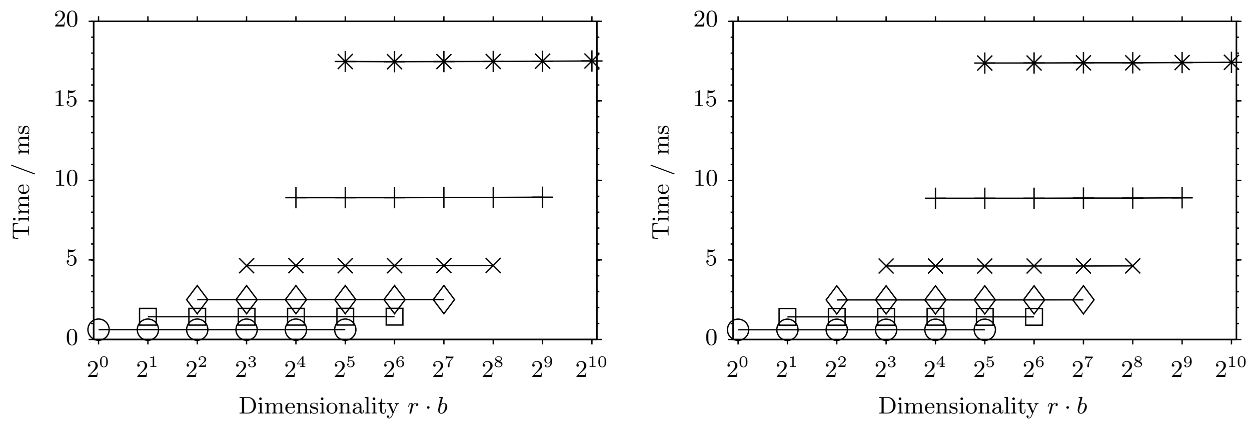
Figure 6.9.: Average computing times of signature functions independent of r and b. The black line depicts the mean computing time obtained from averaging over times obtained for the parameter combinations of r and b. The shaded area depicts the range between minimum and maximum computing time resulting from averaging over the computing times obtained for different choices of r and b. The signatures $s_{\texttt{mm}}$, $s_{\texttt{mv}}$, $s_{\texttt{ms}}$, $s_{\texttt{mk}}$, $s_{\texttt{mmv}}$, $s_{\texttt{msk}}$, $s_{\texttt{mmvs}}$, $s_{\texttt{mvsk}}$ and $s_{\texttt{mmvsk}}$ rely on the same function for computing the statistical moments and are therefore subsumed under the label $s_{\texttt{mx}}$.

computing times obtained from 1000 repeated time measurements.

To get a first impression of the results, we pooled the data obtained for each signature function and computed an average, which is independent of the choice of the parameters r and b. The results of this first evaluation step are presented in figure 6.9. The average computing time of each signature is depicted by black circles, signatures are sorted in increasing order depending on the average, and the shaded area marks the range between minimum and maximum computing time. Signature functions based on statistical moments (equations (6.6) to (6.14)) are summarized under the label $s_{\texttt{mx}}$. The 9 different signatures are internally computed by the same C/C++ function which —independent of the statistical moments contained in the signature— always computes *all four* moments. Hence, the computing times of different signatures are identical and only depend on the number of rings r.

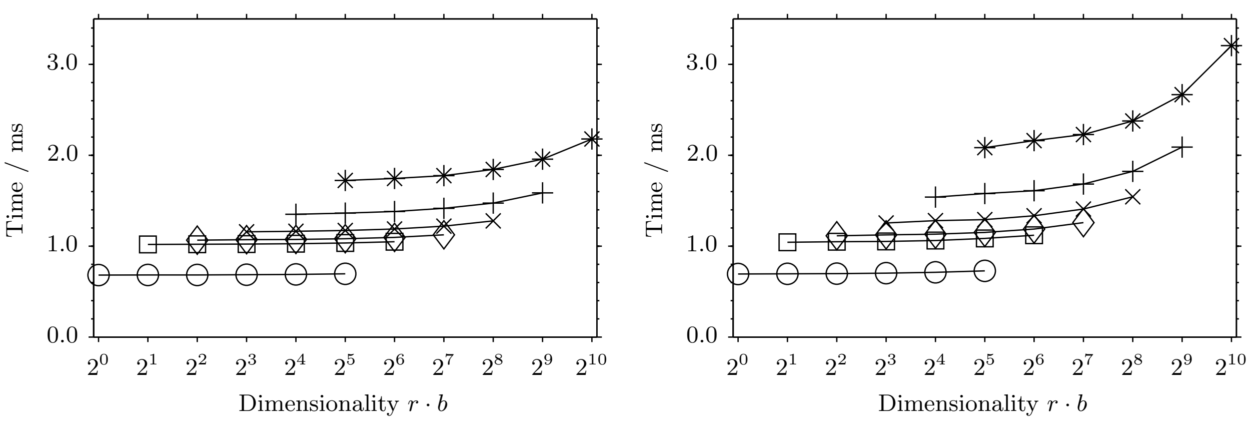
The results reveal that the signatures based on statistical moments can be computed most efficiently. Computing times range from 0.82 ms to 1.65 ms with an average of 1.20 ms. With mean computing times of 1.22 ms and 1.41 ms, the histogram-based signatures $s_{\texttt{hist}}$ and $s_{\texttt{chist}}$ require a comparable effort. The computing times of these signatures vary in the ranges between 0.68 ms and 2.18 ms for $s_{\texttt{hist}}$ and between 0.69 ms and 3.2 ms for $s_{\texttt{chist}}$. Thus, the signature's dimensionality only has a small influence on the time required to compute the signature. For the $s_{\texttt{cog}}$ signature, we obtained an average computing time of 3.08 ms. As the signature only depends on the number of rings r, the computing time only varies from 2.72 ms to 3.41 ms. The Fourier-based signatures $s_{\texttt{afc}}$ and $s_{\texttt{zafc}}$ both obtain mean computing times of 5.9 ms and therefore require the largest computational effort. Depending on the specific choice of r and b, the minimum and maximum computing times range between 0.61 ms to 17.42 ms for $s_{\texttt{afc}}$ and between 0.61 ms and 17.52 ms for $s_{\texttt{zafc}}$.

As the best loop-closure detection results in sections 6.3.2.1 and 6.3.2.3 were obtained for Fourier-based and histogram-based signatures, we analyze these signatures in more detail. The results are shown in figures 6.10 and 6.11 and visualize the computing time against the signature's overall dimensionality $r \cdot b$. The number of rings is coded by different markers (see the captions of figures 6.10 and 6.11 for explanation); along the line obtained for each value of r, the number b of Fourier coefficients or histogram bins increases from left to right according to equation (6.51). Figure 6.10 depicts the results obtained for the Fourier-based signatures $s_{\texttt{afc}}$ and $s_{\texttt{zafc}}$. The results for both signatures are identical because the computation of the signatures relies on the same C/C++ and Tcl functions. For up to 8 rings (×), the signature can be computed in less than 5 ms. For $r = 16$ (+) and $r = 32$ ($*$), the computing time increases to 8.9 ms and 17.4 ms, respectively. The required computing time only depends linearly on the number of rings r and is independent of the number of coefficients b. This is due to the used algorithm for the discrete Fourier transformation, which computes all Fourier coefficients and not only the first b coefficients required for the signature. For the dimensionalities obtained by several combinations of r and b, the signatures computed with



(1) $\mathtt{s_{afc}}$ **(2)** $\mathtt{s_{zafc}}$

Figure 6.10.: Computing times of Fourier-based signatures depending on the number of rings r and the number of Fourier coefficients b. Subfigures (1) and (2) depict the results for the signatures with and without DC-component, respectively. The number of rings r is coded by different markers as follows: (∘) $r = 1$, (□) $r = 2$, (⋄) $r = 4$, (×) $r = 8$, (+) $r = 16$, and (∗) $r = 32$.



(1) $\mathtt{s_{afc}}$ **(2)** $\mathtt{s_{zafc}}$

Figure 6.11.: Computing time of histogram-based signatures depending on the number of rings r and the number of Fourier coefficients b. Subfigures (1) and (2) depict the results for the signatures based on intensity histograms and cumulative intensity histograms, respectively. The number of rings r is coded by different markers as follows: (∘) $r = 1$, (□) $r = 2$, (⋄) $r = 4$, (×) $r = 8$, (+) $r = 16$, and (∗) $r = 32$.

larger r require more effort.

The histogram-based signatures $\mathtt{s_{hist}}$ and $\mathtt{s_{chist}}$ can be computed with considerably less effort (figure 6.11). For the signature $\mathtt{s_{hist}}$ based on gray-value histograms, the computation with up to 16 rings (+) takes less than 1.3 ms. For $r = 32$ rings (∗), most of the signatures can be computed in less than 2 ms; only the signature with $b = 32$ histogram bins takes 2.2 ms. In comparison to the $\mathtt{s_{hist}}$ signature, the signature $\mathtt{s_{chist}}$ takes slightly longer to compute. In this case, the computation of signatures with up to 4 rings (×) takes less than 1.3 ms. Signatures with up to 16 rings (+) require less than 2.1 ms of computing time. Computing signatures with $r = 32$ (∗) takes between 2.0 ms and 3.2 ms. For both signatures, the effort depends mainly on the number of rings r and to a small extent on the number of histogram bins b. The latter aspect is probably due to normalizing the histograms to unity and —in case of the $\mathtt{s_{chist}}$ signature— to cumulating the histogram.

We conclude that the statistical signatures and the histogram-based signatures can be computed

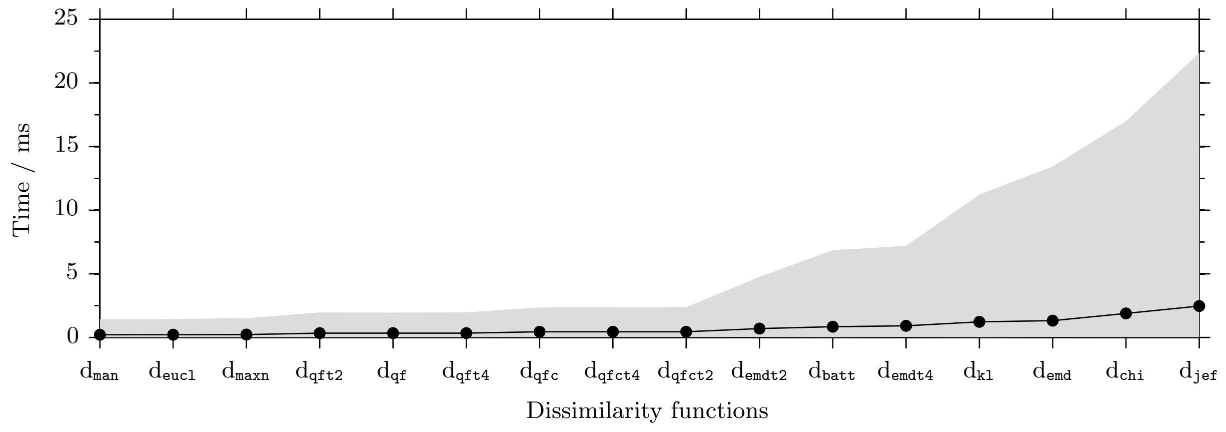


Figure 6.12.: Average computating time of dissimilarity functions independent of r and b. The black line depicts for each dissimilarity function the mean computing time required to compare two signatures. The range between minimum and maximum computing times is visualized by the shaded area.

faster than the Fourier-based signatures. For the histogram-based signatures, the number of bins b and the number of rings r only has a small influence on the resulting computing time. This is in contrast to the Fourier-based signatures, for which the resulting computing times is solely influenced by the number of rings r. However, by choosing a small or a moderate number of rings r, computing times of up to 3 ms can be obtained. For these cases, the effort required to compute Fourier-based signatures is comparable to the effort of histogram-based signatures with a large dimensionality. The computing time of the Fourier-based signatures can further be reduced by recursively computing the required coefficients as described by Stürzl and Mallot [611].

6.3.2.5. Computational Aspects of Dissimilarity Functions

The objective of this section is to analyze the computational effort of comparing two signatures with the tested dissimilarity functions. We therefore follow the experimental procedure described in section 6.3.1.3. For each dissimilarity functions, we determine computing times for comparing two signatures with dimensionalities for all possible combinations of r and b resulting from equations (6.50) and (6.51). The given computing times are median values computed from 1000 repeated measurements.

To get a first impression of the results, we computed for each signature function the average computing time over all possible parameter combinations of r and b. The results are depicted in figure 6.12 with the average computing time marked by black circles. Average computing times were sorted in increasing order, and the shaded area marks the range between minimum and maximum computing time for each dissimilarity function. The results reveal that the norm functions d_{man}, d_{eucl}, and d_{maxn} require the least time —even though the used implementation of these functions relies on the Tcl standard library [I93]. The computing times of all three functions vary between 0.01 ms and 1.51 ms with average values of 0.21 ms. The quadratic form functions (d_{qf}, d_{qft2}, and d_{qft4}) and the quadratic χ^2 functions (d_{qfc}, d_{qfct2} and d_{qfct4}) achieve average computing times of 0.34 ms and 0.45 ms, respectively. The maximum computing times of these functions are 2.4 ms. Because the results of these functions are close together, it is hard to draw reliable conclusions which of these functions performs better or whether thresholded variants require less effort than unthresholded variants. The thresholded variants of the earth mover's distance achieve mean computing times of 0.70 ms and 0.91 ms for d_{emdt2} and d_{emdt4}, respectively, and the average computing time for d_{emd} is 1.32 ms. In this case, thresholded variants with a smaller distance threshold t perform better than the unthresholded variant d_{emd}. For all variants, the exact time required for comparison of signatures depends strongly on r and b, and the maximum computing times of the earth-movers distance are

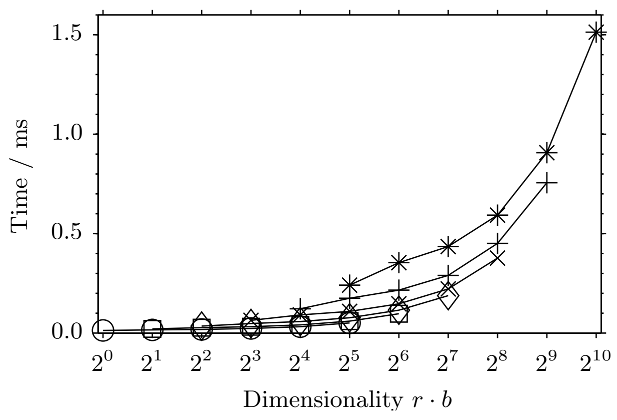


Figure 6.13.: Computing time of maximum norm $\mathrm{d}_{\mathtt{maxn}}$ depending of the signatures' dimensionality $r \cdot b$. The number of rings r is coded by different markers: (∘) $r = 1$, (□) $r = 2$, (⋄) $r = 4$, (×) $r = 8$, (+) $r = 16$, and (∗) $r = 32$.

considerably larger than that of the quadratic form and quadratic χ^2 functions. The bin-by-bin matching functions $\mathrm{d}_{\mathtt{bhat}}$, $\mathrm{d}_{\mathtt{kl}}$, $\mathrm{d}_{\mathtt{chi}}$, and $\mathrm{d}_{\mathtt{jef}}$ perform worst and require in the average 0.84 ms, 1.23 ms, 1.88 ms, and 2.47 ms to compare two signatures. Like for the earth mover's distance, the effort required to compute the bin-by-bin matching functions strongly depends on the signature's dimensionality.

Since we identified the maximum norm $\mathrm{d}_{\mathtt{maxn}}$ as dissimilarity function with a good performance (sections 6.3.2.2 and 6.3.2.3), and as it will be used for the real-robot experiments (section 6.4), we analyze this function in more detail. Figure 6.13 visualizes the computing time required to compare two signatures against the signature's dimensionality. The computation of the norm takes less than 0.5 ms for $r \cdot b \leq 128$ and less than 1.0 ms for $r \cdot b \leq 512$. Comparing two signatures with an overall dimensionality of 1024 requires 1.5 ms. The results show that the computational effort depends linearly on the overall dimensionality $r \cdot b$. For dimensionalities $r \cdot b$ obtained for several combinations of r and b, computing the dissimilarity takes longer if more rings are used.

Due to the prototype implementation of the tested dissimilarity functions, drawing conclusions from this comparison is difficult, because the implementation is based on different frameworks and on code optimized to a different extent. For a well-founded comparison, the measurements should be repeated with a reimplementation of the methods in C/C++ and a comparable level of optimization.

6.3.3. Discussion and Conclusions

In the previous sections 6.3.2.1 to 6.3.2.5 we have presented results of testing the proposed signature functions and dissimilarity functions on image databases. In order to identify promising combinations and parameter settings (i) for a wide range of different environments and (ii) for different illumination conditions, we pooled the results into two test groups and searched for methods with best possible performance for both groups. The test groups are referred to as `day|day` and `day|night` group and contain image databases collected under natural illumination (`day|day`) and cross databases collected during day under natural illumination and during the night under artificial illumination conditions (`day|night`).

Among all tested signatures, statistical signatures can be computed most efficiently. However, their loop-closure detection performance is not as good as the performance of histogram-based and Fourier-based signatures. The histogram-based signatures $\mathrm{s}_{\mathtt{hist}}$ and $\mathrm{s}_{\mathtt{chist}}$ can —even for large dimensionalities— be computed very efficiently. The drawback of the histogram-based signatures is that the specific choice of the number of rings r and the number of histogram bins b has a large influence on the resulting loop-closure detection performance. It seems that the signatures perform better for a larger number of rings r and a relatively small number of histogram bins b. However, for none of the two test groups, the histogram comparison functions with cross-bin matching perform better than standard norm functions. These results contradict our expectations because cross-bin matching functions were designed to tolerate a certain extend of variability in the histograms as it can occur due to changes of the illumination conditions (sections 6.2.2.1 and 6.2.4). We also would not have expected the low robustness against illumination changes of the earth-mover's distance, which was designed to tolerate intensity changes and which is often used as reference in the field of histogram comparison methods (e.g. [508, 509]). Even in combination with cross-bin matching functions, histogram-based signatures are not more tolerant against changes of the illumination than Fourier-based methods.

For the `day|day` group, the Fourier-based signatures $\mathrm{s}_{\mathtt{afc}}$ and $\mathrm{s}_{\mathtt{zafc}}$ achieve very good results for more than 8 dimensions and over a wide range of combinations of r and b. However, their performance is strongly decreased for the `day|night` test group. Although the $\mathrm{s}_{\mathtt{zafc}}$ signatures omitting the first Fourier component representing the average image brightness should theoretically be more robust against changes of the illumination (section 6.2.4), their performance is not superior to that of the signature $\mathrm{s}_{\mathtt{afc}}$ containing all components. We consider this result to be an effect of the illumination changes simulated by the cross-database experiments, which cannot be described by a simple change of the overall image brightness. With the current implementation, the complexity of the Fourier-based signatures depends linearly on the number of rings r. For a moderate number of rings, Fourier-based signatures take only slightly more time to compute than histogram-based signatures. Due to these reasons, we will solely test the Fourier-based signature $\mathrm{s}_{\mathtt{afc}}$ in the real-robot experiments described in section 6.4. Regarding the dissimilarity functions, the experiments revealed that the choice of the dissimilarity function does not have a large influence on the performance of loop-closure detection with Fourier-based signatures. For these purposes, the maximum norm $\mathrm{d}_{\mathtt{maxn}}$ will be used for real-robot experiments.

The main drawback of signature-based loop-closure detection is the methods' little robustness against changes of the illumination. For all tested signature and dissimilarity functions, the loop-closure detection performance strongly impaired for the `day|night` group. With the large proportion of misclassifications and the large spatial distance between expected match and computed match, the methods do not allow for reliable loop-closure detection. These findings also diminish the validity of the results obtained for assessing the robustness against perceptual aliasing and perceptual variability. It seems that combinations of signatures with a higher dimensionality are slightly more robust against changes of the illumination than combinations with a lower dimensionality. However, the considered cross-database experiments contain strong changes of the illumination which are resulting from collecting the images during completely different illumination conditions. We therefore performed real-robot experiments under more realistic illumination conditions in order to reveal whether the tested methods are sufficient (section 6.4).

Besides the accuracy of loop-closure detection and the robustness of the proposed methods against changes of the illumination, their applicability on a real robot is an essential aspect. For the experimental conditions under which the database experiments were conducted, computing a single 128-dimensional Fourier-based signature $\mathrm{s}_{\mathtt{afc}}$ takes 5 ms and that comparing two such signatures requires 0.5 ms (sections 6.3.2.4 and 6.3.2.5). Even for large apartments, we do expect that —even for large-scale environments— 80 to 100 signature comparisons are sufficient for loop-closures (see section 5.6.2 for a motivation of this number). Loop closures have to be detected in the time between two consecutive snapshots are added to the robot's dense topo-metric map. Assuming the robot to move with 10 cm/s and that new snapshots are added to the robot's map every 10 cm (these are similar conditions than used for the real-robot experiments described in sections 4.4.1 and 6.4), there is at most 1 s of time for loop-closure detection. In case only the current signature has to be derived and all other signatures are stored in the map, almost 2000 signature comparisons can be computed. If the signatures of the mapped places are not stored, but have to be recomputed, only around 180 comparisons are possible. Thus, we consider signature-based loop-closure detection to be applicable on a real cleaning robot even if such a robot is equipped with less computational power than the computer used for the database experiments and even if other tasks, such as visual

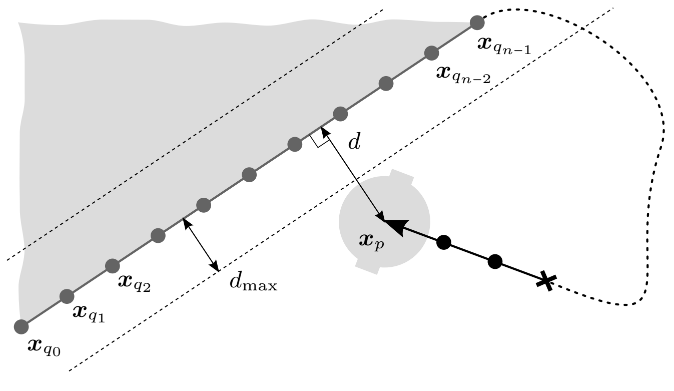


Figure 6.14.: Experimental procedure for real-robot experiments. First, a reference lane of snapshots (dark gray circles) is recorded, which is used as "virtual fence". After the robot is released from the start position (black cross) it moves towards the reference lane. While moving, it continuously compares the current signatures to the signatures stored along the reference lane (filled circles). In case a loop closure is detected, the robot stops, and its final distance d to the reference lane is analyzed. For further details see sections 6.4.1.1 and 6.4.1.3.

homing or position estimation, have to be executed in parallel.

6.4. Real-Robot Experiments

In the previous section, the combination of the Fourier-based signature with DC-component $\mathrm{s}_{\mathtt{afc}}$ in conjunction with the maximum norm $\mathrm{d}_{\mathtt{maxn}}$ as dissimilarity function was identified as a promising approach for visual loop-closure detection being applicable on a real robot. In order to evaluate the combination under more realistic conditions, real-robot experiments were performed which are described in this section. The common experimental procedure, the data evaluation methods, and the robot setup are described in section 6.4.1. Section 6.4.2 describes the results of the experiments; the section ends with a discussion of the obtained results (section 6.4.3).

6.4.1. Methods

This section first describes the experimental procedure (section 6.4.1.1), robot setup and the experimental framework used to perform the robot experiments (section 6.4.1.2), and the data evaluation methods (section 6.4.1.3).

6.4.1.1. Experimental Procedure

The experimental procedure is visualized in figure 6.14. At the beginning of each experiment, a reference lane is recorded (figure 6.14; black line with black circles). The reference lane is assumed to be the outer border of an already cleaned segment (light-gray area) and will, during the course of the experiment, serve as a "virtual fence". While moving along the reference lane (dark gray), the robot acquires omnidirectional images every 10 cm (this corresponds to recording an image per second). The positions of image acquisition $\boldsymbol{x}_i$ ($i = 0, 1, 2, \ldots$) are in figure 6.14 marked by filled circles. The distance between two snapshots is estimated based on the robot's on-board odometry. The images are unfolded and the signatures $\boldsymbol{q}_i$ are computed, which are stored in a purely topological map without position information. After the end of the lane (decided by the operator), the robot is manually moved away from the reference lane to an arbitrary start position (black cross) and is released heading towards the reference lane. While approaching the reference lane, the robot again takes images every 10 cm (filled circles), unfolds them, and derives the current signature $\boldsymbol{p}$. The signature $\boldsymbol{p}$ is compared to all signatures $\boldsymbol{q}_i$ stored in the map (i.e. along the reference lane) resulting in n dissimilarity values

$$\ell_i = \mathrm{d}(\boldsymbol{p}, \boldsymbol{q}_i). \tag{6.54}$$

The robot stops if

$$\hat{\ell} = \min_{i=0}^{n-1} \ell_i \tag{6.55}$$

falls below the decision threshold ℓ_t:

$$\hat{\ell} < \ell_t. \tag{6.56}$$

The experimental framework (section 6.4.1.2) allows to perform two types of experiments referred to as *short-term* and *long-term* experiments. For the *short-term* experiments, 10 test trials were performed immediately after recording the reference lane. As a single short-term experiment takes approximately 10 min, this type of experiments therefore only covers abrupt changes of the illumination (table 6.9). See section 3.2.3.2 for a more detailed discussion on changes of the illumination. In contrast, *long-term* experiments reuse the reference lane recorded at a previous point in time. Depending on the time between recording the reference lane and testing, the method's robustness against medium-term changes and against daytime changes of the illumination can be assessed. For the long-term experiments, we recorded the reference lane and released the robot from 5 different start positions. The testing phase began immediately after recording the reference lane and was repeated for 7 times with approximately 10 min between trials. Thus, our experiments cover changes of the illumination as occurring within 1 h (table 6.11). This is the time expected (i) to clean a relatively large workspace and (ii) to be the maximum battery life before recharging is required. Unfortunately, we cannot show panoramic images visualizing the illumination changes which occurred during both experiments because the image data got during a system update.

6.4.1.2. Implementation and Robot Setup

The experimental procedure (section 6.4.1.1) was mainly implemented by Oliver Schlüter as a student assistant supervised by Lorenz Hillen. This included (i) a reimplementation in C/C++ of the signature computation and comparison and (ii) the integration into the Tcl framework for cleaning-robot control developed by Prof. Dr. Ralf Möller to easily allow for extensions of these strategies in future work. Minor modifications on the experimental procedure, the parameter tuning, and the experiments were carried out by Lorenz Hillen.

For the experiments, our experimental cleaning robot was used (section 4.4.3 and figure 4.11). As setup to acquire omnidirectional images, a panoramic annular lens (Tateyama PAL-S25G3817-27C, [D16]) in combination with a CCD camera (IDS Imaging UI-2220SE-M, [D9]) were used (see figure 3.7.2 for a photo of the setup). To keep the average image brightness of the camera images constant, a PID controller was used to adjust the exposure time. Camera images were unfolded on the robot's on-board computer (IEI Technology PM-US15W-Z530-R10 with an Intel Atom Z530 CPU, [D10]) following the approach described in Krzykawski [345]. Unfolding included a histogram equalization followed by a low-pass filtering with a binomial filter of kernel size 11. The size of the resulting images was 360 × 48 pixels covering a vertical field of view from 0° to 38° above the horizon. The client-server architecture for the communication between the robot and the host computer as well as unfolding and preprocessing on panoramic images was implemented by Martin Krzykawski. The unfolded camera images were transmitted via wireless-network connection to an external host computer, which was used to compute, store, and compare image signatures (figure 4.12). In our case, a laptop [D4] equipped with a quad-core Intel Core i7 920XM and 4 GB of RAM was used as external host computer.

For all experiments, $r = 8$ rings and $b = 12$ Fourier coefficients were used resulting in 96-dimensional signatures. Due to the smaller field of view and the smaller image size, we also used lower-dimensional signatures. The robot was moving with 10 cm/s, and the decision threshold was for all experiments set to $\ell_t = 0.4$.

6.4.1.3. Evaluation

During the experiment, the robot's spatial position $\boldsymbol{x}$ is tracked using the active tracking system (figure 6.15). This position estimate is solely used for data evaluation and allows for determining the robot's current distance d to the reference lane. For this purpose, a single line is fitted through the positions $\boldsymbol{x}_i$ by linear regression. This also reduces the influence of outliers in the position estimates obtained by the active tracking system. However, if the reference lane is slightly curved due to systematic errors of the robot's odometry (section 4.4), a piecewise linear interpolation of the reference lane would more accurately fit its true shape. For data evaluation, we analyze the percentages of the following outcomes of the experiment depending on the robot's distance d_s to the reference lane after stopping because $\ell_{\min}$ fell below the threshold ℓ_t (figure 6.16):

`correct`: In this case the robot stopped *on* the reference lane with $|d_s| \leq d_{\max}$. Although stopping the robot on the lane means a small amount of repeated coverage, it avoids uncleaned areas. For our experiments, we chose $d_{\max} = 15\,\mathrm{cm}$, which corresponds to approximately half the diameter of the used robot and the inter-lane distance used to cover the workspace by meandering lanes (section 4.4).

`gap`: The robot stopped *in front of* the lane and $|d_s| > d_{\max}$. If the robot stops before reaching the lane, an uncleaned area remains. It should be the main goal of all implemented cleaning strategies to avoid such uncleaned areas.

`overlap`: The robot crossed the lane and stops *behind* the lane with $|d_s| > d_{\max}$. With respect to cleaning strategies, this outcome would increase the proportion of repeated coverage therefore reducing the robot's cleaning efficiency. Nevertheless, full coverage of the robot's workspace could still be achieved.

For the case that the robot does not stop because $\ell_{\min}$ does not fall below the threshold ℓ_t, a fourth case is considered:

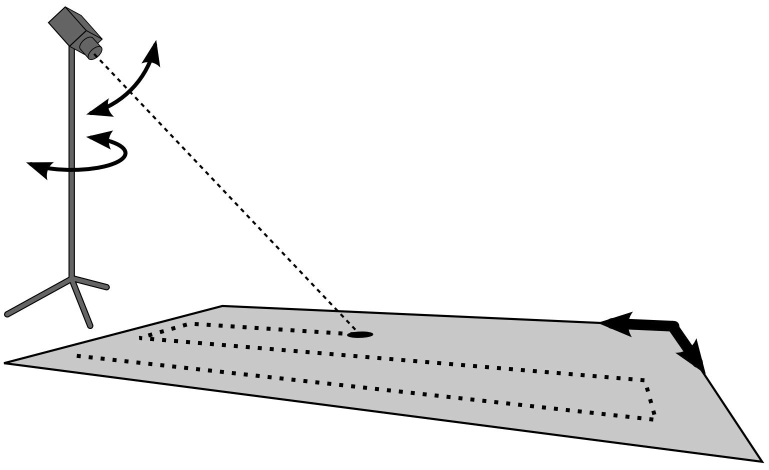
`failed`: The robot's distance to the reference lane is continuously computed, and the robot is automatically stopped if the distance increases $2d_{\max}$. This case not only results in a large portion of repeated coverage but also indicates that loop-closure detection failed. Thus, it increases the probability that an inconsistent map of the environment is maintained and that the map cannot be corrected in further processing steps.

This evaluation was done online while performing the experiment.

To quantify the changes of the illumination occurring during the course of the experiments, we use the global radiation R, measuring the directed and diffuse components of the solar radiation. The measurement values of R are provided online by the weather station located on the campus of Bielefeld University and are 10 min averages updated every 2 min [I97]. Abrupt changes of the illumination are not visible in the measurements due to averaging. Furthermore, the measures of R do not exactly represent the illumination conditions because (i) we cannot measure the global radiation inside our lab and because (ii) the weather station is approximately 400 m away from the lab of the Bielefeld Computer Engineering group, in which the experiments were performed. However, we feel that monitoring the global radiation R gives at least a hint how the illumination conditions vary during the experiments. For a more detailed discussion on the global radiation, the reader is referred to Ahrens [1].

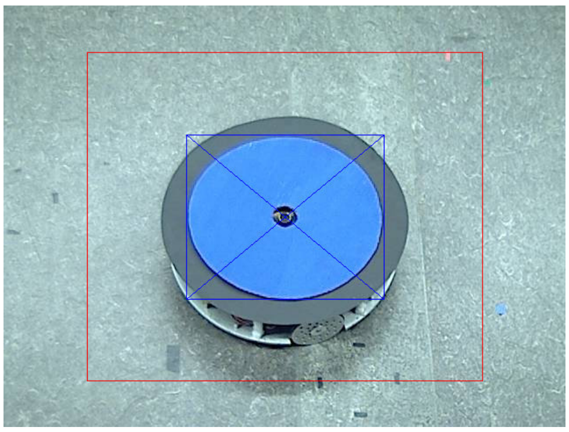
6.4.2. Results

Based on the experimental procedure explained in the previous section, 12 short-term and 5 long-term experiments were performed. Their results are described in sections 6.4.2.1 and 6.4.2.2, respectively. A video showing an experiment similar to the short-term experiments is available for download [S5].

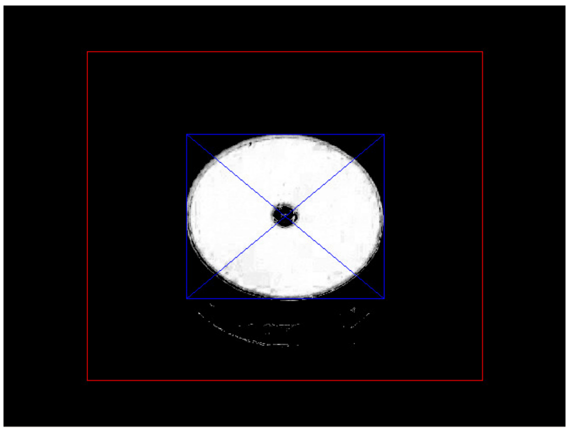


(1) Sketch of the principle

(2) Camera and pan-tilt unit. Photo by Lorenz Hillen.



(3) Camera image while tracking



(4) Probabilistic color segmentation

Figure 6.15.: Active visual tracking system. Subfigure (1) visualizes the principles of the tracking system. A camera is mounted on a pan-tilt unit, which is moved such that the robot keeps centered in the camera image. From the joint angles of the pan-tilt unit and the robot's position within the camera image, an estimate of the robot's position (black circle) in the experimental area (light-gray area) is computed. Subfigure (2) is a photo of the setup consisting of a pan-tilt unit (Directed Perception PTU-D46.17-5 [D6]) mounted on a tripod (Manfrotto 055XBPro with Manfrotto 496RC2 ball head [I66, I67]) and the camera (camera: Axis 211 W [D3]; lens: Tamron 13VG550ASII [D15]). Subfigure (3) shows the camera image while tracking. The robot's tracking target (in the shown case a blue disk), is detected in the camera image by color segmentation. To facilitate color segmentation, the camera's color segmentation is oversaturated by setting the corresponding camera parameter to the maximum value. For the pixels inside the search region (red rectangle), the color dissimilarity to the reference color is computed. Subfigure (4) depicts the dissimilarity information with white and black coding identical and dissimilar pixels, respectively. The dissimilarity information is used to track the target, to repeatedly compute its center (blue cross) by applying the cam-shift algorithm (Bradski [66]), to adjust the size of the search window (red rectangle), and to adaptively adjust the reference color to the current illumination conditions. The computed center position is used to generate a motion command for the pan-tilt unit to keep the robot centered in the camera image. The deviation of the center position from the image center is fused with the joint angles of the pan-tilt unit to estimate the robot's position based on triangulation. For further details, please refer to the project thesis by Daniel Venjakob [668], who also implemented the system. With an image resolution of 640 × 480 pixels, tracking is possible at approximately 5 images per second and an accuracy of 1 cm. In its current implementation, the system can only estimate the robot's position but not its orientation. The advantages of the active tracking system over the passive system (figure 4.15 and section 4.4.5) are its portability and ease of calibration. Figure requires color printing.

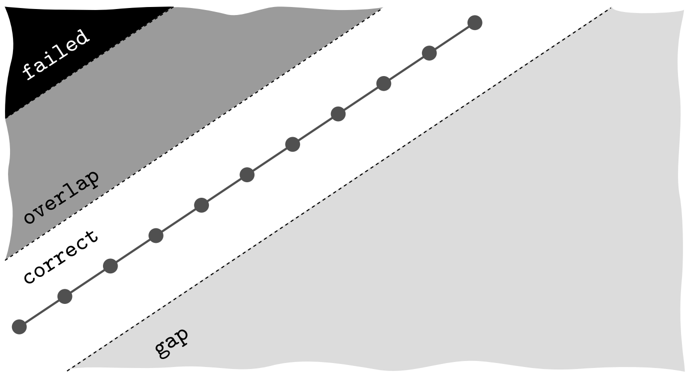


Figure 6.16.: Possible outcomes of real-robot experiments. When the robot stopped, its distance to the reference lane (dark gray line with circles indicating snapshot positions) is evaluated and grouped into four different categories: `gap` if the robot stops before the reference lane (light-gray area), `correct` if it stops on or close to the lane (white), `overlap` if it stops behind the reference lane (mid gray), and `failed` if the robot does not detect loop closures (black). For further details see sections 6.4.1.1 and 6.4.1.3.

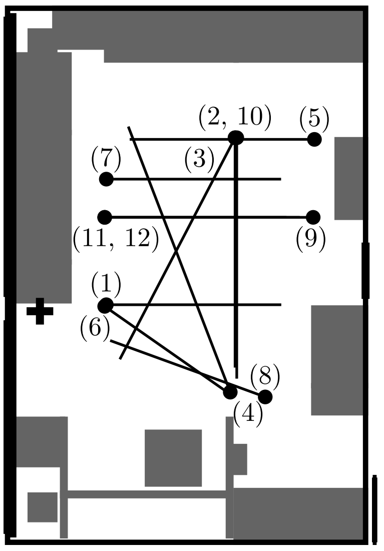
Table 6.9.: Experimental conditions of short-term experiments. Experiments 1 to 8 were conducted under nearly constant illumination conditions; experiments 9 to 12 include rapidly changing conditions due to strong natural or artificial changes of the illumination. The global radiation R measures the illumination conditions outside the main building of Bielefeld University. The measures are 10 min averages provided by [I97]. For some experiments, the values of R are not available due to server errors; the values are only given for experiments which were performed under natural or under natural with additional artificial illumination.

			R / W/m^2		
Exp.	Date	Time	Begin	End	Weather and illumination conditions
1	2011-09-30	15:35	—	—	Light-proof curtain closed, artificial illumination only.
2	2011-09-30	15:45	n.a.	510	Overcast. Additional artificial illumination
3	2011-09-30	16:00	510	n.a.	Overcast. Diffuse natural illumination without artificial illumination.
4	2011-10-04	11:55	224	118	Overcast, rainy. Diffuse natural illumination.
5	2012-02-13	11:30	129	160	Overcast. Diffuse natural illumination.
6	2012-02-13	13:25	143	113	Overcast. Diffuse natural illumination.
7	2012-02-13	17:05	47	41	Overcast. Sunset at 17:35. Diffuse natural illumination only.
8	2012-02-16	12:20	148	161	Overcast. Diffuse natural illumination.
9	2012-02-17	14:00	325	143	Partly cloudy with clear spells. Illumination rapidly darkened during recording of reference lane. Diffuse illumination during tests.
10	2012-02-20	14:40	245	206	Mostly overcast with a clear spell. Diffuse illumination, rapidly brightened during recording of reference lane. Diffuse illumination during tests.
11	2012-02-21	14:30	145	133	Overcast. During recording of reference lane additional artificial illumination. Artificial illumination turned off for tests.
12	2012-02-21	14:55	115	114	Overcast. Natural and artificial illumination. During recording of reference lane, the neon lamps closer to the door were switched on. For tests, the neon lamps close to the door were switched off and the ones close to the windows were turned on. See figure D.1 for positions of neon lamps.

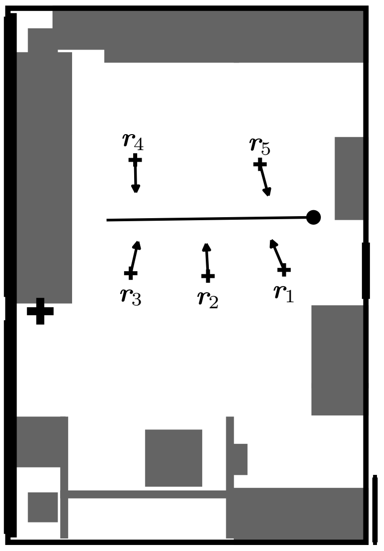
6.4.2.1. Short-Term Experiments

For the short-term experiments, the test trials are performed immediately after recording the reference lane. A total of 12 short-term experiments with 10 test trials per experiment were conducted. The start positions of the reference lane and the reference lanes are visualized in figure 6.17.1. The start positions and orientations of the test trials were randomly chosen and are documented in figure D.1.1. Table 6.9 summarizes the weather and illumination conditions under which experiments were performed.

As experiments 1 to 8 were collected under nearly constant illumination conditions and as experiments 9 to 12 were performed under strong changes of the illumination, we separate the evaluation of the experiments into two groups. For the first group (table 6.10.1), the robot stopped correctly on the lane in approximately 86 % of the cases. In 10 % of all tests, the robot stopped



(1) Short-term experiments



(2) Long-term experiments

Figure 6.17.: Experimental conditions of short- and long-term robot experiments. For the short-term experiments (subfigure (1)), the reference lanes of the 12 experiments are shown with start positions marked by a filled circle. Subfigure (2) visualizes the reference lanes of the long-term experiments (start position again marked by a black circle) together with the five tested start positions. The five start positions $\boldsymbol{r}_i$ are marked by black crosses; the release *directions* (but not the full lanes) are depicted by black arrows. The measuring line at the right bottom of the figures shows the length of 1 m.

Table 6.10.: Results of short-term experiments. The two subtables subsume experiments conducted under nearly constant illumination conditions (subtable (1)) and under strongly varying illumination conditions (subtable (2)). The outcomes of the experiments and the performance measures given in the table are introduced in section 6.4.1.3 and figure 6.16.

(1) Nearly constant illumination conditions

	Outcome			
Exp.	`correct`/ %	`gap`/ %	`overlap`/ %	`failed`/ %
1	70.0	20.0	0.0	10.0
2	80.0	20.0	0.0	0.0
3	90.0	10.0	0.0	0.0
4	70.0	30.0	0.0	0.0
5	80.0	0.0	0.0	20.0
6	100.0	0.0	0.0	0.0
7	100.0	0.0	0.0	0.0
8	100.0	0.0	0.0	0.0
Avg.:	86.25	10.00	0.00	3.75

(2) Strong illumination changes

	Outcome			
Exp.	`correct`/ %	`gap`/ %	`overlap`/ %	`failed`/ %
9	10.0	0.0	0.0	90.0
10	0.0	0.0	0.0	100.0
11	0.0	0.0	0.0	100.0
12	0.0	0.0	0.0	100.0
Avg.:	2.50	0.0	0.0	97.50

Table 6.11.: Experimental conditions of long-term experiments. The global radiation R measures the illumination conditions outside the main building of Bielefeld University. The measures are 10 min averages provided by [I97]. The values of R are only given for experiments which were performed under natural illumination.

			R / W/m^2				
Exp.	Date	Time	Begin	Min	Max	End	Weather and illumination conditions
1	2012-02-13	17:30	—	—	—	—	Light-proof curtain closed. Constant artificial illumination only.
2	2012-02-14	08:40	45	45	76	66	Overcast with light snow. Diffuse natural illumination with clearly visible brightness changes.
3	2012-02-14	16:50	65	34	65	34	Overcast. Experiment conducted during dusk (sunset 17:37). Diffuse illumination getting considerably darker. For the last trial, the illumination was not sufficient for tracking the robot.
4	2012-02-20	09:00	80	80	135	135	Partly cloudy with clear spells. Overcast at the end of the experiment. Mostly diffuse illumination.
5	2012-02-20	14:55	174	174	342	296	Mostly sunny with some clouds. Illumination conditions ranging from diffuse to directed illumination.

too early (`gap`); the tests failed in approximately 4 % of the cases. Thus, with a decision threshold of $\ell_t = 0.4$ and nearly constant illumination, reliable loop-closure detection is possible. With 10 % of the trials resulting in the outcome `gap`, the threshold could further be decreased because with the current parameter choice, the decision threshold seems in some cases to be reached before the robot reaches the reference lane. However, this would probably also increase the proportion of `overlap` or `failed`. None of the tests resulted in the outcome `overlap`. We therefore suspect that the dissimilarities increase again once the robot crossed the reference lane.

For experiments 9 to 12 performed under strong changes of the illumination (table 6.10.2), the tests failed in 97.5 % of the cases. The robot only stopped correctly in 2.5 % of the 40 trials; the outcomes `gap` and `overlap` did not occur. Thus, reliable loop-closure detection under strong changes of the illumination is not possible with the used combination $s_{\texttt{afc}}$, $d_{\texttt{maxn}}$, $r = 8$, and $b = 12$, or with the specific choice of the decision threshold ℓ_t. Increasing the decision threshold ℓ_t could improve the proportion of correct detections under strong illumination changes, but it would also increase the proportion of `gap` situations while reducing the proportion of `correct` detections for constant illumination conditions. With the experiments described in the following section, we further investigated the robustness of the proposed methods against changes of the illumination occurring over a longer period in time.

6.4.2.2. Long-Term Experiments

A total of five long-term experiments were conducted under the illumination and weather conditions summarized in table 6.11. After recording the reference lane, we performed tests from five different release positions (figure 6.17.2) in our lab (figure D.1.2). For all experiments, identical start positions of the reference lane and of the test trials were used. We decided for start positions of the test runs (figure 6.17.2; crosses) being relatively close to the reference lane because this allows us to perform the five test runs of each trial shortly after each other. By this means, all five test runs of a trial could be conducted under the same illumination conditions. Furthermore, it left us enough time to change batteries and restart the robot between two consecutive trials without delaying the start of the next trials because a single experiment could not be performed within one battery charge. However, with start positions in the vicinity of the reference lane, local minima causing false-positive loop-closure detections are extremely unlikely. Such false positives would cause the robot to stop before reaching the reference lane, therefore increasing the proportion of the outcome `gap`. For *all* experiments conducted in our lab —including not only the presented experiments

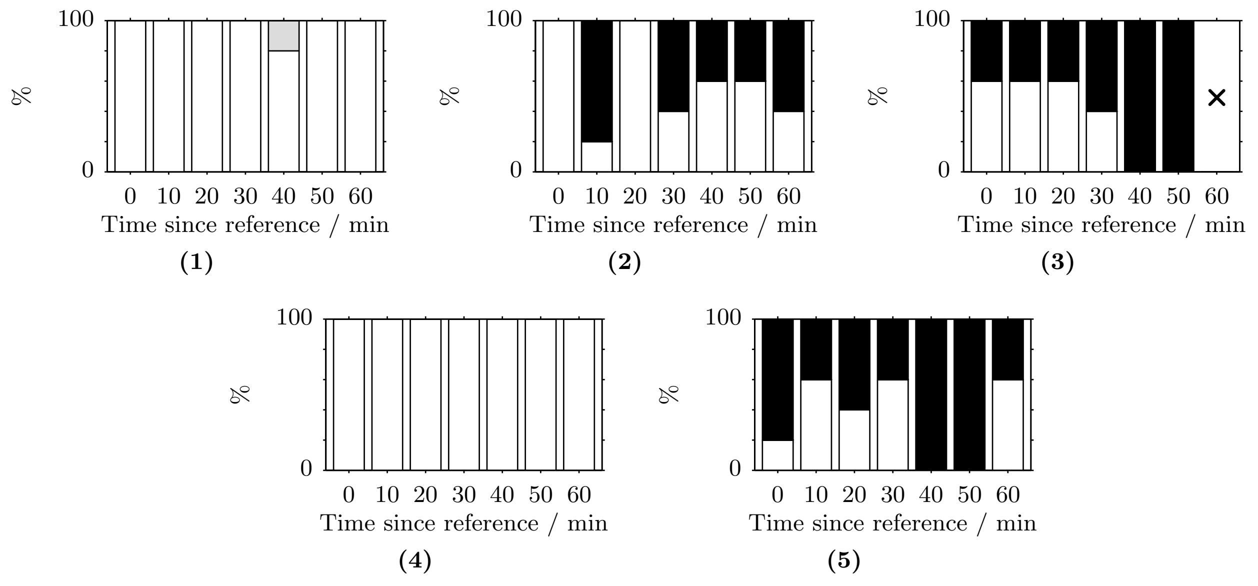


Figure 6.18.: Results of long-term experiments. Subfigures (1) to (5) depict the results for the five different experiments. Each bar of the subfigures corresponds to a trial (trials are not numbered but rather the time since recording the reference lane is given) of the corresponding experiment with the outcomes coded by different gray values as follows: white: `correct`; light-gray: `gap`; black: `failed`. The outcome `overlap` did not occur during the experiments. For experiment (3), tracking the robot during trial with an offset of 60 min was no longer possible because of the illumination inside the lab being too dark. The corresponding trial is therefore marked by a black cross.

but also preliminary test while developing the experimental framework and adjusting the decision threshold— we never observed false positives due to local minima. This effect is most likely due to the environmental properties of our lab. Thus, we do not expect that choosing start positions farther apart from the reference lane had influenced our results, but for repeating the experiments in a different room, this issue has to be considered for defining the start positions of the test runs.

The first test trial was started immediately after recording the reference lane. The following six test trials were performed with approximately 10 min of time between trials. Thus, experiments were performed over 1 h. The resulting trajectories are visualized in figure D.3. During the course of the experiments, the frequencies of the four outcomes `correct`, `gap`, `overlap`, and `failed` were computed as described in section 6.4.1.3. The results of the experiments are shown in figure 6.18. The figures show for each test trial the proportion of the outcomes `correct` (white), `gap` (light gray), and `failed` (black). Like for the short-term experiments, the outcome `overlap` did not occur.

Experiment 1 (figure 6.18.1) is the control experiment performed under constant and artificial illumination with the lab's light-proof curtain being closed. In all but one cases, the robot stopped correctly on the lane; in a single case it stopped too early (`gap`). All remaining experiments were performed under natural illumination conditions with the lab's curtain and sun-blind being open. The results of the trials for experiment 2 (figure 6.18.2) vary: the experiment was started approximately 1 h after sunrise under visible changes of the brightness inside the lab. For trials with 0 and 20 min offset, the robot stopped correctly in all cases, whereas for the trial with 10 min offset it only stopped correctly in one case; in the other cases the outcome `failed` occurred. The remaining cases yielded the outcome `correct` in approximately half of the cases. In all case the robot did not stop correctly, the experiment failed. Experiment 3 (figure 6.18.3) was started approximately 50 min before sunset. Thus, the illumination in the lab darkened during the course of the experiment. For trial 7 (60 min offset), the active tracking system was no longer capable of tracking the robot (the trial is in figure 6.18.3 marked by a cross). Over time, the proportion of correct loop-closure detections decreases: for the first three trials (0 to 20 min), the robot stops correctly in three of five cases, for the trial with 30 min offset it stops correctly in two cases and for the trials with offsets of 40 min and 50 min it did not stop correctly. As experiment 4 was performed (figure 6.18.4) under relatively small changes of the illumination, the robot correctly detects loop-closures in all cases. During recording the reference lane of experiment 5 (figure 6.18.5), strong abrupt changes of the illumination occurred. The results therefore show a large variability and only a weak performance. Correct detection in three of five cases was obtained for the trials with offset of 10 min, 30 min, and 60 min; two of five cases were correctly detected for the trial with an offset of 20 min. The trial performed immediately after recording the reference lane yielded only a single correct detection, and for the trials with offsets of 40 min and 50 min all test runs failed. All other triales failed.

The results clearly show that reliable loop-closure detection over a longer period of time is only possible if the illumination conditions are nearly constant; for stronger changes of the illumination, the performance decreases. In case the illumination conditions changed during the course of the experiment, the variability of illumination conditions is reflected in the variability of correct loop-closure detections. Like already mentioned in the discussion of the short-term experiments (section 6.4.2.1), replacing the threshold-based decision by a more sophisticated approach could increase the method's performance. We therefore conclude that the used cross-databases (`day|night` test group) are more realistic than we expected. Furthermore, we expect to be realistic scenarios even more challenging then the laboratory situation tested here. In the following section, the results obtained by the short-term and long-term robot experiments will be discussed.

6.4.3. Discussion and Conclusions

To assess the proposed methods' robustness against changes of the illumination under more realistic conditions, real-robot experiments were performed. Therefore, we recorded a reference lane which was during the test trials used as "virtual fence": while approaching the reference lane from various release positions, the current signature was compared to the signatures derived from the snapshots taken along the reference lane. The robot stopped if the minimum dissimilarity between the current signature and the reference signatures fell below the decision threshold ℓ_t. For performance evaluation, the proportion of the following four cases was analyzed: (i) correct loop-closure detection (`correct`), (ii) stopping too early (`gap`), (iii) stopping to late (`overlap`), and (iv) stopping because the robot's spatial distance to the reference lane increased the threshold $d_{\max}$ (`failed`). Following this procedure, two types of experiments covering different types of illumination changes were performed. Short-term experiments mainly cover abrupt changes of the illumination because test trials were performed immediately after recording the reference lane. Long-term experiments were performed over 1 h with repeated tests of the reference lane recorded at the beginning of the experiment. Thus, these experiments cover abrupt, medium-term, and, to some extent, also daytime changes of the illumination.

Regarding the robustness against changes of the illumination, we think that the currently used method is not yet applicable on a real cleaning robot. The results of both types of experiments clearly reflect the variability of the illumination conditions: on the one hand side, reliable loop-closure detection is possible in case of small or moderate changes of the illumination influencing mainly the overall image intensities. In this case the used exposure controller can reduce the influence of illumination changes already at the time of image acquisition. On the other hand side, abrupt changes of the illumination or changes of the illumination mainly causing local intensity changes currently cause the loop-closure detection to fail. Increasing the tolerance against changes of the illumination should therefore be the primary goal of future work.

With respect to computational complexity, the results of the experiments show that the proposed method is applicable on a real robot. Although the number of compared snapshots will probably be larger if the method is integrated into more complex cleaning strategies (section 6.6) and a real cleaning robot will use an on-board computer with less computational power, we expect that there will be sufficient time for the required signature comparisons.

6.5. Overall Discussion and Conclusions

In this chapter, we have approached the loop-closure detection problem by deriving global image signatures from the entire image and comparing these signatures instead of a pixel-by-pixel comparison of entire images as for holistic loop-closure detection methods (chapter 5). As the signature dimensionality is considerably smaller than the number of pixels in the image, the method allows for much more efficient image comparisons. In case parsimonious signature functions are chosen, the effort of deriving the signature is negligible, and signatures can be stored in the robot's map. Thus, repeated applications of the signature functions are avoided. This approach is suitable for topological maps with and without metrical position information.

From screening a wide range of signature and dissimilarity functions, the Fourier-based signatures with ($s_{\mathtt{afc}}$) or without DC component ($s_{\mathtt{zafc}}$) and the signatures based on intensity histograms ($s_{\mathtt{hist}}$) or cumulative intensity histograms $s_{\mathtt{chist}}$ have been identified as promising signatures (section 6.3.2.1). As dissimilarity function, we prefer the maximum norm $d_{\mathtt{maxn}}$ because the tested histogram-comparison functions —both with and without cross-bin-matching— did not achieve a better loop-closure detection performance but are computationally more complex. Especially the earth mover's distance $d_{\mathtt{emd}}$ and its thresholded variants $d_{\mathtt{emdt2}}$ and $d_{\mathtt{emdt4}}$ did not yield better results than the other distance functions —although they are widely used as baseline method for image-retrieval experiments (e.g. [132, 508, 509]). These results are surprising because the cross-bin matching functions ($d_{\mathtt{qf}}$, $d_{\mathtt{qfc}}$, and $d_{\mathtt{emd}}$ and their threshold counterparts $d_{\mathtt{qft}}$, $d_{\mathtt{qfct}}$, and $d_{\mathtt{emdt}}$) were designed to compensate for variations of the histograms resulting from changes of the illumination. Since we expected these functions to allow for reliable loop-closure detection even under strong illumination changes as simulated by our cross-database tests, we did not test image preprocessing methods as we did for holistic loop-closure detection. This issue will be addressed in future work (section 6.6.1).

Further evaluations of the Fourier-based and histogram-based signatures in combination with the maximum norm $d_{\mathtt{maxn}}$ (section 6.3.2.2) revealed that the performance of the histogram-based signatures depends stronger on the choice of the parameters than the performance of the Fourier-based signatures. The latter allow —at least for the test group containing smaller changes of the illumination— for a very accurate loop-closure detection. Furthermore, the Fourier-based signatures in conjunction with the maximum norm $d_{\mathtt{maxn}}$ as dissimilarity function offer a wide range of parameter combinations with perfect or close to perfect robustness against perceptual aliasing and perceptual variability (section 6.3.2.3). For the test group containing strong changes of the illumination, the performance regarding loop-closure detection (sections 6.3.2.1 and 6.3.2.2) and robustness against perceptual aliasing and perceptual variability (section 6.3.2.3) is considerably reduced. Especially for the latter evaluation, identifying suitable combinations of signature and dissimilarity functions and optimal parameters is hardly possible, because results only vary over a small and comparably worse performance range. According to the results obtained for our database experiments, the signature $s_{\mathtt{zafc}}$ without the first Fourier coefficient (DC component) is less robust against changes of the illumination than the signature $s_{\mathtt{afc}}$ containing the first coefficient. These results are surprising: because the signature $s_{\mathtt{zafc}}$ is independent of the first Fourier component, which represents the average image brightness, we would have expected that it is more robust against illumination changes than $s_{\mathtt{afc}}$. A possible explanation of the result is that the changes of the illumination simulated by our cross-database experiments cannot be described by a simple change of the overall image brightness (equation (5.52)). Due to the obtained results, we prefer the $s_{\mathtt{afc}}$ signature over the $s_{\mathtt{zafc}}$ signature. For this signature, the best results were obtained for 16- to 128-dimensional signatures. In case of 128-dimensional signatures, this means that the image information is reduced to approximately 5 % of its original size. The computation of a signature with such a dimensionality takes approximately 5 ms with the current implementation and the currently used hardware; the comparison of two signatures requires at most 0.5 ms. To this end, we conclude that the proposed methods are applicable on a real cleaning robot and that the comparison is efficient enough to compare a larger number of signatures as required for navigation based on dense topological maps.

This conclusion was also validated by real-robot experiments (section 6.4). For the real-robot experiments, a set of reference snapshots was collected in regular distances while the robot was moving along a straight cleaning lane. These reference snapshots were during the experiment used as the border of an already cleaned segment (metaphorically speaking they form a “virtual fence”). For testing, the robot was released heading towards the reference lane and was supposed to stop on the lane thus avoiding both gaps between cleaned segments and larger areas of repeated coverage. Therefore, the robot’s current signature was repeatedly compared to the signatures stored along the reference lane. When the resulting dissimilarity fell below a threshold, the robot stopped and its spatial distance to the reference lane was analyzed. Following this procedure, two types of experiments were conducted: short-term experiments covering changes of the illumination as occurring over a period of 10 min, and long-term experiments covering changes as occurring over 1 h. The results of both types of experiments (sections 6.4.2.1 and 6.4.2.2) reveal that reliable and accurate loop-closure detection is possible for constant illumination conditions or moderate changes of the illumination. For stronger changes of the illumination or changes influencing only parts of the images’ appearance, the proposed methods fail. As such changes occurred independent (i) of the time of day when experiments were performed and (ii) of the duration of the experiments, we conclude that the proposed methods are not yet applicable on a real cleaning robot. Approaches to increase the robustness should therefore be the primary goal of future work (section 6.6).

In comparison to holistic loop-closure detection methods relying on global image comparisons (chapter 5), the methods described in this chapter are clearly superior with respect to computational complexity and storage requirements. Storing the lower-dimensional signature together with the snapshot in the robot’s topological map requires considerably less storage than storing the snapshot together with a preprocessed variant of it. A further advantage over holistic methods is that the proposed image signatures are invariant against the robot’s orientation. Thus, signature-based methods do not need the compass step requiring most of the computing time of holistic methods. Regarding the robustness against changes of the illumination and against perceptual aliasing or perceptual variability, the best methods relying on image signatures perform significantly worse than the best methods relying on global image comparisons. Although much information is lost by the transformation to the lower-dimensional signature, we expect that the future working directions can improve the robustness of signature-based methods.

6.6. Future Working Directions

As already pointed out, the primary goal for future work should be to increase the methods’ robustness against changes of the illumination (section 6.6.1). Once the robustness against illumination changes is increased, we expect the methods to be a valuable sub-system of more complex cleaning strategies (section 6.6.2). Signature-based approaches also offer interesting properties for topological or topometric map-building (section 6.6.3) and for modeling insect navigation behavior (section 6.6.4). Beyond that, the proposed methods could also be used as parameter-based homing methods (section 3.5.2.2). As extensive tests in [177, 537] have shown that parameter-based homing methods are not competitive to other homing methods, we do not further discuss this possible line of research.

6.6.1. Increasing the Robustness Against Illumination Changes

In chapter 5 we could show that image preprocessing can considerably increase the robustness against changes of the illumination. This principle is also applicable for signature-based loop-closure detection, and we expect that it can increase the method's robustness against illumination changes. In addition, signatures computed from edge information could be tested because edge information is often less dependent on the illumination than intensity information (e.g. [644]). One starting point is the signature proposed by Gu and Chen [261], which is a histogram of edge-detector responses computed by summation over the image columns of a panoramic image preprocessed with a vertical edge detector (in this particular case the Sobel filter; equation (5.9)). In contrast to the signatures tested in this chapter, the signature by [261] is orientation-dependent and therefore requires the application of a compass. To achieve rotation invariance, [261] suggest to align the histogram w.r.t. the strongest edge response, i.e. with the largest entry of the histogram. In principle we expect this signature to exhibit a certain tolerance against illumination changes, but its robustness strongly depends on the quality of the compass method, which is difficult to judge without testing the signature. Two images acquired at identical or nearby positions in space can only be correctly aligned if their strongest edges result from the same object in space. If the visual appearance of the images is too different (e.g. because of illumination changes), the strongest edge could result from different objects. In this case, the compass and hence also loop-closure detection will fail.

6.6.2. Integration into a More Complex Control Scheme

In case the robustness against changes of the illumination can be increased, the methods should be integrated into the framework for cleaning strategies currently developed by several members of the Bielefeld Computer Engineering group (section 7.3.2). The framework allows for systematically cleaning complex workspaces by combining several segments covered by meandering lanes. To reliably detect loop-closures is a prerequisite for achieving a consistent map of the robot's environment thus avoiding both uncleaned areas and large proportions of repeated coverage. Therefore, every snapshot added to the topological map has to be compared to snapshots at the border of already cleaned areas. If loop-closures are detected, the spatial relations between cleaning segments can be corrected in order to increase the quality of the underlying topological map. Such an approach is closely related to hybrid mapping and hierarchical SLAM (sections 3.6.2 and 7.3.3). By comparing the robot's current signature to all signatures stored in the map (and not only to the signatures at the borders of cleaning segments), the proposed methods could also be applied to solve the kidnapped-robot problem, i.e. the problem that the robot has to relocalize after being manually displaced by the user (section 7.3.2).

6.6.3. Signature-Based Approaches to Topological Mapping

The proposed signature-based loop-closure detection can also be applied for mapping and localization in combination with topological or topo-metric maps with a signature-based place representation (sections 3.6.3.2 and 3.6.4.1). Especially for systems with limited computational power and limited storage capacities, the proposed methods allow for efficient navigation strategies and offer a possibility to cover larger workspaces.

Beyond purely map-building, clusters computed in the signature space can be exploited to derive spatial information about the robot's workspace. Each cluster represents a region of the robot's workspace; doors or passages between regions should be detectable by transitions between clusters. Such approaches are similar to the methods described by [475, 545, 725, 726]. For room segmentation, the methods establish correspondences based on two-view stereo geometry or based on a feature-based dissimilarity measure incorporating the relation of correspondences and the total number of features (Zivkovic, Bakker, and Kröse [725] and Zivkovic, Booij, and Kröse [726]). Rottmann et al. [545] proposes to segment rooms by applying an AdaBoost classifier (e.g. [196, 671]) on the sensor data obtained from a laser range finder and a monocular camera. Clustering techniques could also be used to reduce the number of images contained in a topological map or for hierarchical localization (section 3.6.2). In both cases a "centroid signature" is computed representing an entire set of similar images in its spatial vicinity. For clustering, standard methods (e.g. k-means or hierarchical clustering; textbooks: [46, 148, 533]), vector quantization methods (e.g. neural gas [197, 413]), or optimal graph cuts [725, 726] could be used. To obtain topo-metric maps by reintroducing metrical information (section 3.6.3.2), algorithms related to multi-dimensional scaling (textbooks: [148, 533]) could be used to optimize the spatial positions of the snapshots according to pairwise dissimilarities computed in the higher-dimensional signature space. Beyond classical multi-dimensional scaling, extensions such as locally linear embedding (e.g. [546]), relevance learning vector quantization (e.g. [274]), or isometric feature mapping (e.g. [626]) could be used. Alternatives to such methods are mass-spring models as already considered in related work (sections 3.3.1.3, 3.6.3.2 and 3.6.4.1). This branch therefore opens many possibilities for developing visual SLAM methods relying on a topo-metric map and a signature-based representation of places (section 3.6.3.2). For testing the proposed methods, the image databases summarized in table 3.3 applicable for building *sparse* topological maps could be used, which we think are better suited for this problem than the *dense* databases used in this chapter.

6.6.4. Biological Modeling

The proposed methods can also be applied to tackle interesting working directions in the field of visual insect navigation. There is an ongoing debate how insects make use of the visual navigation and how image information is represented (sections 3.3.3.3, 3.5.3.1 and 3.6.5.2). In Graham and Cheng [248], parts of the visual panorama were masked, and the ants' homing performance was compared for masking different regions. The experiments revealed that the lower portion of the visible panorama is necessary and sufficient for insect homing. Following the idea of this experiment, one could mask out one or more rings from the signatures by not considering the corresponding subdescriptors for computing dissimilarities (equation (6.48)). By systematically analyzing how masking out certain rings influences the loop-closure detection performance, one could identify the most relevant parts of the image. From a biological perspective, if applied to biologically plausible input data (i.e. images collected in the insect's habitat and at the level of the insect's eyes, Zeil [717]) this technique should be capable of identifying the image regions containing the most relevant information. By a similar approach, the transition between ground and sky was identified as most relevant image region for desert ants (Basten and Mallot [31]). For robotics, this approach can help to further improve the methods' complexity by only comparing relevant subdescriptors or to increase the loop-closure detection performance by weighting subdescriptors depending on their relevance or reliability. In this working direction, we see the concepts of rings as a tool to identify relevant image regions, but we do not presuppose that something comparable is also found in insect brains.

7. Overall Summary, Discussion, and Outlook

Section 7.1 of this chapter summarizes the proposed navigation strategies and the obtained experimental results of (chapters 4 to 6). Section 7.2 briefly discusses the main result and the main conclusions; future working directions are described in section 7.3.
Several aspects of future work are currently pursued or were already tackled by other members of our research group working on the same project, namely by David Fleer, Michael Horst, Janina de Jong, Martin Krzykawski, Prof. Dr. Ralf Möller, and Andreas Stöckel. The corresponding paragraphs and figures are marked accordingly.

7.1. Overall Summary

The objective of this section is to give a condensed summary of the main results and conclusions described in more detail in section 7.2. This thesis describes basic navigation strategies for a domestic floor-cleaning robot using omnidirectional vision as primary sensory information. Our particular application poses two main challenges for the developed navigation strategies (section 2.2). First, the robot has to efficiently cover its entire workspace while avoiding both uncleaned and repeatedly covered areas. Second, the developed strategies have to be in line with the sensory and hardware equipment of the robot. Since domestic cleaning robots are consumer goods, they can only be equipped with a small number of cheap sensors and limited computational power.

Our navigation methods rely on dense topo-metric maps as spatial representation, on local visual homing to estimate spatial relations, and on meandering lanes as basic motion strategy. As dense topo-metric maps and local visual homing are concepts which are strongly influenced from biological research on visual insect navigation, these two building blocks of our methods can be considered to be bio-inspired. Bio-inspired navigation methods have proven to be parsimonious yet robust and powerful solutions [189, 453, 677, 678]. The experiments described in this thesis successfully prove the feasibility of the these building blocks for cleaning-robot control. We are furthermore of the opinion that they are feasible for a potential consumer product. Our navigation strategies are the first application of these building blocks for navigation of domestic floor cleaning robots, and —together with the navigation strategies for a mobile service robot operating in a do-it-yourself store [255–258, 327, 338–340]— the first application of omnidirectional vision to a complex real-world task. Since a complete control scheme for a full-fledged cleaning robot is beyond the scope of this thesis, we focused on *trajectory control and mapping* and on *visual detection of already cleaned areas*. We consider these two substrategies as most important because they can be used as basis for more complex strategies such as detecting and approaching uncleaned areas.

Trajectory Control and Mapping

Regarding trajectory control and mapping (chapter 4) we propose a mostly vision-based controller, which is used to guide the robot along parallel and meandering lanes while concurrently mapping the robot's environment. We use a dense topo-metric map as spatial representation, which characterizes places with the visual information, in our case a panoramic image, perceived at the place. Thus, we use former robot positions as landmarks for estimating spatial relations and obtain a straightforward representation of already cleaned areas. The map is extended while the robot moves along a cleaning lane. The visual information stored along its current lane is used on the subsequent lane to estimate the robot's current distance to its previous lane. Using a dense topo-metric map has proven its value because (i) it can be easily built from the available sensor information, (ii) it offers the fine spatial resolution required for accurate cleaning-robot navigation, and (iii) it scales well with the size of both the robot's environment and the already cleaned area.

For estimating the robot's current distance to the previous lane, we rely on a parsimonious lane-distance estimator fusing bearing information from local visual homing and odometry information. By minimizing the deviation between the robot's true and desired distance to the previous lane, the robot can be guided along meandering and parallel lanes. Despite the simple control algorithm, which does not rely on an elaborated position estimation technique such as particle- or Kalman-filtering, we obtain a total coverage of approximately 90% (80% of the workspace are covered exactly once, 10% are covered twice, while 10% remain uncleaned). In contrast to the vast majority of related approaches (section 3.6.3.2), we do not estimate the robot's full pose but only perform partial pose estimation. By solely computing the robot's current distance to the previous lane and its current orientation, we compute exactly the information which is sufficient and necessary to solve the task: with only one of these estimates lacking, the robot could no longer fulfill its task. The idea behind partial pose estimation is to save unnecessary computations. Our method is computationally less complex than comparable methods maintaining a full pose estimate by particle-filtering (Möller et al. [457]) or Kalman-filtering (Jong [314]). However, since our method lacks a full pose estimate, it does not allow to follow arbitrary trajectories specified in world coordinates. It therefore complicates further navigation strategies such as approaching uncleaned areas or returning to the robot's charging station. To this end, we will in future work abandon partial pose estimation in favor of the particle-filter method by Möller et al. [457], which in addition produces straighter cleaning lanes.

Visual Detection of Already Cleaned Areas

We investigated two approaches for visual detection of already cleaned areas: *holistic* and *signature-based* loop-closure detection methods. Both methods compare the robot's currently perceived camera image with the images stored in the map. Holistic methods (chapter 5) compare the entire images pixel-by-pixel, whereas signature-based methods (chapter 6) derive low-dimensional image signatures from the entire images and compare these instead. For both types of methods, we systematically tested a wide range of different approaches to evaluate their loop-closure detection accuracy and their robustness against illumination changes. The results of our experiments reveal that holistic measures allow for very accurate loop-closure detection even under strong changes of the illumination. The drawbacks of these methods are (i) that they are computationally demanding, (ii) that they require the application of a visual compass (see below), and (iii) that they are —at least with the current rapid-prototyping implementation— not applicable on a cleaning robot which is supposed to clean large rooms. In contrast, signature-based approaches are very efficient. They allow for accurate loop-closure detection under (nearly) constant illumination conditions, but their performance is strongly impaired if strong changes of the illumination occur. Nevertheless, we favor the application of signature-based approaches for application on a real cleaning robot, even though additional effort is needed to increase their tolerance against illumination changes. An ideal loop-closure method would fuse the efficiency of signature-based methods with the robustness of holistic methods.

A difficulty of both holistic and signature-based loop-closure detection methods is to find an appropriate decision threshold for classification whether or not the compared images are identical. Each place node stored in the map is surrounded by a region in space, and images acquired within this region are by our loop-closure detection methods considered to be acquired at the same position in space than the image associated with the place node. The region's extent directly depends on the decision threshold used for loop-closure detection: a small threshold leads to a small region, whereas a large threshold leads to a large region. If the threshold is chosen too large, loop-closures are detected although the robot is a certain distance away from the place node. For cleaning, this will result in gaps between segments. If the threshold is chosen too small, detecting the border of an already cleaned segment may fail, and repeated coverage could occur. To this end, our particular application requires small decision thresholds and a fine spatial resolution for precise and accurate loop-closure detection avoiding both uncleaned and repeatedly covered areas.

Holistic loop-closure detection methods require the considered images to be aligned w.r.t. a common reference direction, a prerequisite which can be fulfilled by applying a visual compass method prior to loop-closure detection. The standard compass method by Zeil, Hoffmann, and Chahl [718] requires a step-wise shift of one of the images and a repeated evaluation of the comparison function. We proposed an accelerated compass variant which operates in the Fourier domain and can estimate the compass shift from a single image comparison. Under constant illumination conditions, our method computes accurate compass estimates, but its performance drastically decreases under changing illumination conditions. In this case, the standard method achieves more accurate results, but it is computationally less efficient than the accelerated compass variant.

The substrategies proposed in this thesis will in future work be used as basis for more elaborated substrategies which are required to make the robot capable of completely covering complex-shaped workspaces. This is achieved by decomposing the entire workspace into several segments of meandering and parallel lanes as they are obtained from our trajectory controller. More elaborated substrategies will use the dense topo-metric map for (i) identifying uncleaned areas, (ii) approaching them, and (iii) for orientation along the first lane of a new cleaning segment. Further directions of future research include hierarchical mapping based on the decomposition into cleaning segments and using the omnidirectional vision setup as multi-purpose sensor for further purposes such as obstacle detection or user interaction.

7.2. Overall Discussion

The navigation strategies proposed in this thesis are a first step towards a full-fledged cleaning robot capable of autonomously and reliably covering complex-shaped workspaces such as entire apartments. Among all strategies needed for such a robot, we focused on two essential substrategies: (i) visual trajectory control and mapping (chapter 4), and (ii) visual detection of already cleaned areas (chapters 5 and 6). These two substrategies were implemented based on the following four building blocks (section 1.2): (i) omnidirectional vision as primary sensory information, (ii) meandering lanes as motion strategy, (iii) dense topo-metric map as spatial representation, and (iv) local visual homing to estimate spatial relations. As far as we know, these building blocks have been for the first time applied for navigation of a domestic floor-cleaning robot. The proposed navigation strategies were experimentally tested with our custom-built research platform. The promising results obtained throughout chapters 4 to 5 clearly reveal the applicability of the proposed strategies and of the underlying building blocks. The particular strengths and weaknesses of our methods have already been discussed in earlier chapters. Here, we rather focus on aspects which are relevant for several of the considered substrategies. These include aspects of map consistency (section 7.2.1), the relation of the proposed work to research from the field of spatial cognition (section 7.2.2), working towards a prototype product (section 7.2.3), and alternative applications for the proposed methods (section 7.2.4).

7.2.1. Local Map Consistency vs. Global Map Consistency

The navigation strategies proposed in this work rely on a dense topo-metric map storing the already cleaned parts of the robot's environment. The map is built by the proposed trajectory controller (chapter 4) and can be understood as a locally consistent submap of the entire workspace. If several cleaning segments are combined, the entire workspace is covered by several locally consistent submaps. For the navigation strategies considered in this thesis, the map built by this means is the basis for guiding the robot along meandering cleaning lanes and for detecting already cleaned ares (chapters 5 and 6). At the current state of our work, we keep position estimates attached to the place nodes constant after adding the place node to the map. We therefore avoid the computational effort required by SLAM applications (section 3.6) for subsequent position updates. Instead of enforcing global consistency by SLAM techniques, our future research aims (i) at combining several segments of locally consistent cleaning lanes and (ii) at achieving global consistency between cleaning segments by reliable and accurate loop-closure detection. With respect to future work, interesting questions regarding hierarchical mapping arise from the decomposition of the robot's workspace into several cleaning segments (section 7.3.3). We expect that this line of research will develop interesting and parsimonious alternatives to currently existing SLAM methods.

7.2.2. Relation of the Proposed Work to Spatial Cognition

The field of spatial cognition embraces research on navigation capabilities of animals and humans. Local visual homing and —to a minor extent— also dense topo-metric maps are influenced by the snapshot hypothesis of vision-based insect navigation which states that places can be characterized by the visual information perceived at the place. We consider our navigation methods to be influenced or inspired from biological concepts even though in our concrete case technical solutions are applied (rather than biologically plausible algorithms) to solve an engineering problem. Our triangulation controller uses the min-warping method (section 4.4.2), which was by MÖLLER [456] reformulated to a biologically plausible homing method. Bio-inspired approaches have proven to be parsimonious yet efficient and powerful navigation strategies [189, 453, 677, 678]. As outlined in sections 3.3.3.3, 3.4.3.2, 3.5.3.1 and 3.6.5.2, the question how animals use visual information to represent places, to approach places, or to follow route is subject of ongoing research. In particular our work on signature-based loop-closure detection (chapter 5) can also be used as starting point to contribute mathematical models for animal behavior.

7.2.3. Towards a Prototype Product

When choosing a robot platform for our experiments, none of the commercially available robots was suitable for our needs: the platforms with suitable shape and dimensions did not allow for low-level control, and existing cleaning robots were suitable regarding shape and dimensions, but did not allow for low-level control as is required for the work presented in this thesis. Thus, we decided to custom-built a mobile robot platform similar to available cleaning robots. At the current stage of our work, our robot is rather a research platform than a domestic floor-cleaning robot: (i) computations are performed off-board on an external host computer, and (ii) it is equipped with a better omnidirectional vision setup than a potential product. We decided to equip our platform with a better omnidirectional vision sensor than a potential product because this assures that failures of our navigation strategies are due to limitations of the method itself and not to the quality of the available sensor information. While this decision was reasonable at the early state of our work described in this thesis, a more realistic proof of concept has to rely on hardware which can also be used in a potential product. We will therefore work towards an optimized implementation of our navigation methods which can be executed on-board, and we will equip our robot with sensors, which are also suitable for a potential product. A potential product could be equipped with ARM processor technology by Atmel [I13] or with Atom processors by Intel [I49].

The promising results obtained throughout this thesis reveal that omnidirectional vision is an appropriate sensor modality for a floor-cleaning robot. For working towards a product prototype, the currently used omnidirectional vision setup has to be replaced by a setup which is appropriate for the product's target price. The camera technology available on the market has made a tremendous progress which led to tiny cameras with amazing image quality that are available at relatively low costs (e.g. [600]). In contrast to cameras, catadioptric setups or ultra wide-angle lenses are not yet a mass product, and only a few cheap components at the order of 20 € are commercially available. These components are currently a niche product, and we expect that the price will be further reduced once these setups are produced in larger quantities. These include the catadioptric mirrors designed for cell phones [I41, I62] or for a consumer photo camera [I90]. The lens-mirror combination proposed by [605, 607] could be produced at larger quantities by injection molding for similar price.

We furthermore feel that omnidirectional vision has a large potential as multi-purpose sensor, which can also be applied for further purposes beyond the ones currently investigated. Such purposes include obstacle detection, visual tilt detection, or gesture recognition (section 7.3.4). We expect that applying omnidirectional vision for such purposes can replace the dedicated sensors used otherwise, by this means reducing the hardware costs of a potential product.

7.2.4. Alternative Applications for the Proposed Methods

Alternative applications of our methods include all robot tasks, which require a complete coverage of the robot's workspace. The possible applications can be divided into applications for the *consumer market* and for a *professional market*. The level of *consumer robots* includes robot lawn mowers. Similar to first-generation cleaning robots (section 2.2.1.1), most of the lawn-mowing robots currently available on the market (table 2.1.3) rely on random-walk strategies. The Bosch Indego 10 [I1, I16] is currently the only commercially available cleaning robot relying on a systematic movement strategy. To this end, developing a robot lawn mower capable of systematically covering its workspace seems a promising working direction with a large market potential. We think omnidirectional vision is an appropriate sensor for this field, because relying solely on GPS information (review: [149], textbooks: [41, 586]) is not an option: standard GPS for private usage does not achieve the accuracy required for this task, and differential GPS techniques are too expensive for the consumer market. Instead, a mobile lawn mower relying on omnidirectional visual information could exploit the spectral contrast between ultraviolet and green light. This contrast can be used to segment sky from ground [337, 455] and is similar to the concept of the skyline panorama currently discussed in biology (section 3.5.3.1).

On the *professional level*, robots can be equipped with expensive high-quality sensors. Our navigation strategies could also be applied for navigation of professional cleaning robots. However, elaborated solutions for professional floor-cleaning robots exist (table 2.1.2), and we are therefore of the opinion that placing a product on this market is difficult. Current outdoor applications for professional robots building on complete coverage are, among others, harvesting or demining. For such robots, differential GPS is commonly applied (agricultural robots: [43, 152]) and provides an accuracy sufficient for navigation of such robots. A future application for professional robots could be to systematically cover a polluted or hazardous terrain while collecting measuring samples at a fine spatial resolution. Nevertheless, evaluating omnidirectional vision as a cheap alternative to differential GPS would be an interesting and —if successful— very profitable working direction.

7.3. Future Working Directions

This section discusses four major working directions, which we consider to be the most promising directions substantially extending or improving the work proposed in this thesis. These research directions are (i) ideas for *increasing the robustness against dynamic scene changes* (section 7.3.1), (ii) extending the proposed strategies *towards a full-fledged cleaning robot control scheme* (section 7.3.2), (iii) *hierarchical mapping based on the decomposition of the robot's workspace into cleaning segments* (section 7.3.3), and (iv) using *omnidirectional vision as multi-purpose sensor* (section 7.3.4). Besides these four main directions of future research, several minor issues for future work were already described in previous chapters. These are only briefly recapitulated here:

Section 3.7.1.1: Collecting image databases with our active visual tracking system (figure 6.15) under difficult illumination conditions or in dynamically changing environments. This allows for benchmarking different navigation methods under difficult illumination conditions and to improve them if necessary. Since we plan to make the databases available on the internet, other research groups can test their methods with these databases thus allowing benchmarks of methods developed by different research groups.

Section 5.7.1: Optimizing the current implementation of the proposed holistic loop-closure detection methods in order to make them applicable on a real cleaning robot.

Section 5.7.2: Improving the accelerated compass method by eliminating a minor weakness in the currently used method for computing the image dissimilarity of the best match. By reformulating the computation of the overall matching residual we hope to achieve similar loop-closure detection results with the accelerated compass variant than with the standard compass method.

Section 5.7.3: For our accelerated compass variant and for signature-based loop-closure detection we subdivided the panoramic image into rings. This principle could also be applied for the standard compass method by Zeil, Hoffmann, and Chahl [718]. Each ring would than contribute a compass estimate, and all the ring-wise estimates are then fused to an overall compass estimate. By this means, we hope to reduce the influence of erroneous estimates. The approach could lead to a more accurate compass method which is also more tolerant against image disturbances.

Section 5.7.4: By testing image preprocessing techniques and image dissimilarity functions not considered so far, we hope to find alternatives to the tested methods being executable on a robot's on-board computer. Regarding image preprocessing, we plan to test histogram-warping methods as alternative preprocessing technique (e.g. [123, 259, 260]). As image dissimilarity functions, we will test *local image dissimilarity functions* computing the overall image dissimilarity from comparing a set of local image patches (reviews: e.g. [92, 225]) and the illumination-tolerant correlation *sequential correlation measures* recently proposed by Möller [450]. The implementation of both, the new and the already tested measures, will be optimized in order to allow the methods to be executed on the on-board computer of a mobile cleaning robot.

Section 6.6.1: Increasing the robustness of signature-based loop-closure detection methods against illumination changes to be competitive to holistic methods in order to make it competitive to holistic loop-closure detection.

Section 6.6.3: Investigating methods for signature-based mapping methods and for inferring spatial information from the map structure by clustering signatures. By this means, we could, for

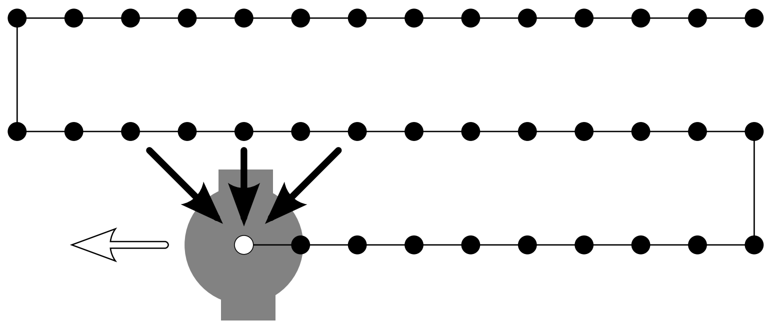


Figure 7.1.: Detecting dynamic scene changes by image prediction. The robot's current view (white circle) can be predicted from neighboring views (thick black arrows). Comparing the prediction and the the really perceived image allows for identifying image regions which are disturbed by dynamic scene changes. The robot's movement direction is indicated by the open arrow.

example, segment the map into different rooms. The obtained segmentation could than be used by further cleaning strategies (section 7.3.2) to optimize the robot's behavior.

Section 6.6.4: Signature-based place representations (section 3.3.1.2) are a parsimonious way to memorize places and to use visual information for navigation. Because of these properties, it is also an interesting hypothesis for how insects use visual information. The proposed signatures and signature-comparison methods can be a basis for further investigating open issues at the intersection between biology and robotics.

Section 7.2.3: In their current implementation, all navigation strategies proposed in this thesis are executed on an external host computer. For future work towards a potential product, it is necessary to reimplement the proposed methods which will be used with a more sophisticated cleaning-robot control scheme in order to make them executable on the robot's on-board computer.

Section 7.2.3: Our custom-built cleaning robot is currently equipped with a better omnidirectional vision setup than a potential product. In the near future, this setup should be replaced by a setup which could also be used for a potential product, and the methods proposed in this work should be extensively tested with this new setup.

7.3.1. Increasing Robustness Against Dynamic Scene Changes

Throughout this thesis, we assumed that the robot is operating in a static environment without dynamic scene changes. The assumption is valid if the robot is cleaning while the user is not at home or in a different room than the robot, but it is clearly violated if humans or pets are moving in the same room as the robot and are visible for the robot. In this case, the dynamic scene changes caused by persons or pets disturb the robot's image and can cause local visual homing or loop-closure detection to fail (section 3.2.3.2).

The robustness of visual navigation methods could be increased by first detecting disturbed areas in the image and not considering them later on for navigation. To identify disturbed image regions, one could predict the robot's current camera image from surrounding images stored in the robot's dense topo-metric map (figure 7.1). For image prediction, we could apply our 2D warping method (Möller, Krzykawski, and Gerstmayr [451] and section 4.4.2) to predict how image columns are shifted and scaled given certain movement parameters of the robot. Disturbed areas are then detected by comparing the predicted image with the actually perceived camera image. The regions identified by this means are then excluded from computing the home vector or from comparing images in the context of loop-closure detection.

Alternatives to the approach outlined above include methods segmenting the optical flow field (textbook: [305, 641], review: [30, 280]) into components due to ego-motion and due to dynamic scene changes. The segmentation is achieved by computing the optical flow between consecutive frames of the camera's video stream. Since the robot's motion between two frames is known (e.g.

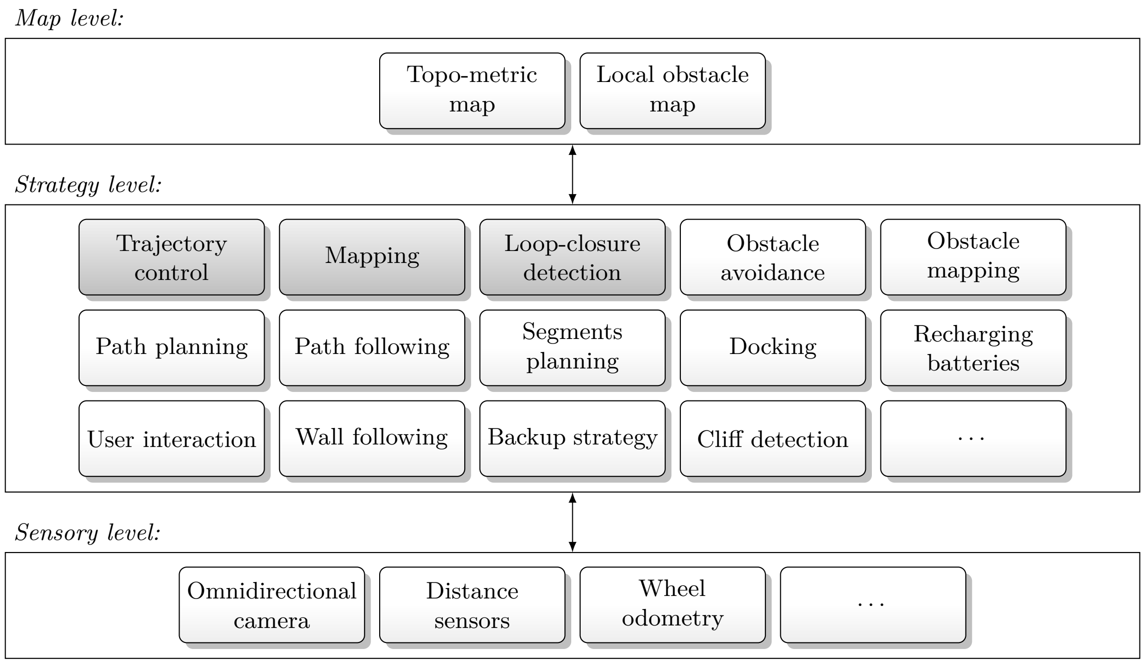


Figure 7.2.: System overview of the control architecture for cleaning complex-shaped workspaces. The shaded subsystems are considered in this thesis.

from odometry), one can predict for each image position the direction (but not the length) of the expected flow vector. In case the predicted and the true direction of the optical flow differ, it is likely that the image shift resulted from an image disturbance. A similar approach is proposed by YOSHIZAKI et al. [713] for obstacle detection based on optical flow.

7.3.2. Towards a Full-Fledged Cleaning Robot Control Scheme[1]

Based on the navigation strategies outlined in chapters 4 to 6, the robot is capable of (i) mapping its environment while concurrently cleaning a single segment with meandering lanes, and (ii) recognizing already cleaned areas. These two substrategies are only essential prerequisites for an autonomous cleaning robot, but are not yet sufficient for completely covering complex-shaped workspaces. For this purpose, several other substrategies (figure 7.2) are required which are not considered in this thesis.

The proposed trajectory controller (chapter 4) covers a rectangular area of the entire workspace by guiding the robot along meandering and parallel lanes. If extending the currently cleaned segment by another parallel lane is not possible because of obstacles or due to reaching an already cleaned area, the system's control architecture (textbook: [41], review: [416]) has to decide where to place the next segment. For this purpose, it consults the map in order to identify border nodes at the border of an already cleaned area facing free space (i.e. place nodes not facing an obstacle or an already cleaned area). In the next step, it has to decide which position is best suited for attaching the next segment. For this purpose, the robot simulates movements over uncleaned terrain and in parallel to the detected border of cleaned areas, and it determines the position of the new segment by searching for the longest possible lane. Then the robot has to plan a path to its start position and has to approach this position. Route planning can be accomplished by standard graph-search

[1]Lorenz Hillen was not involved in implementing the system described in this section, but contributed conceptually and supervised the final theses of Sebastian Ruwisch [550] and Christian Munier [470]. The current system was initially implemented by Prof. Dr. Ralf Möller, and is now further developed by David Fleer and Michael Horst.

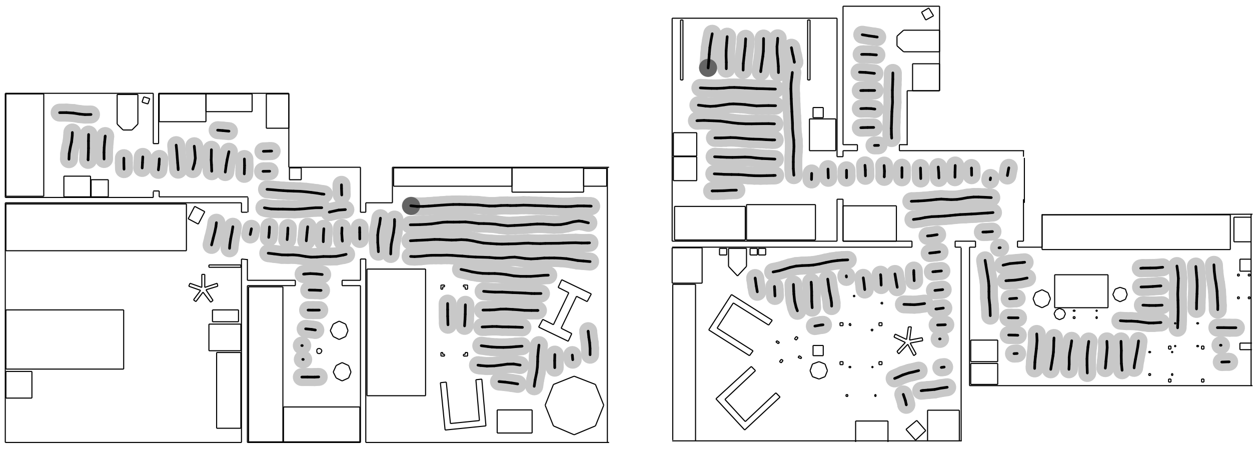

(1) Footprint “Hillen” (2) Footprint “Gerstmayr”

Figure 7.3.: Simulation results for more complex cleaning-robot control scheme. The robot (gray circle) started from the depicted position and covered the largest part of the accessible workspace by several segments of meandering lanes. The robot’s trajectory is depicted by thick black lines; the covered area is depicted in light-gray (assuming the width of the sucction unit to be identical to the robot’s diameter, i.e. 30 cm). The trajectories were in this example obtained from the particle-filter method by Möller et al. [457]. The environment (walls and furniture only) was modeled after two real apartments, namely “Hillen” (subfigure (1)) and “Gerstmayr” (subfigure (2)), each about 50 m^2 in size. The results were obtained in cooperation with Martin Krzykawski by applying the cleaning strategies developed by Prof. Dr. Ralf Möller.

algorithms (textbook: [118]), and local visual homing can be applied for local navigation between intermediate snapshots of the route. After reaching the start position, the robot starts a new cleaning segment and uses border nodes of the previous segment for position estimation and for keeping the segment’s first lane straight. This procedure is repeated until the robot’s workspace is completely covered.

Besides the subsystems for detecting uncleaned areas and for path planning and following mentioned above, a cleaning robot has to include several other substrategies. These include among others (i) approaching its docking station after cleaning or for recharging batteries, (ii) backup navigation strategies if the homing-based trajectory controller fails (e.g. underneath beds if the camera image cannot provide enough visual information), and (iii) the detection of particular situations such as getting stuck (e.g. underneath obstacles or at carpet borders) or approaching stairs. A further aspect not considered in this thesis is the robot’s user interface allowing the user to turn the cleaning process on or off, to select the cleaning mode, or to guide the robot to a certain position it is supposed to clean. Figure 7.2 visualizes a possible system architecture of a full-fledged cleaning robot, and figure 7.3 depicts the results of two simulated experiments in floor plans of real apartments obtained with such a control scheme.

Such a full-fledged control system can not only operate on the camera images and the position estimates attached to each place node of the dense topo-metric map, but it can also add further task-relevant sensor information acquired when visiting the place and use this information at a later point in time to control the robot. Examples of such information are, among others, local obstacle information, tilt information, or camera parameters. The available information could, for example, be used to only select reliable snapshots for taking the bearing by excluding tilted snapshots or by excluding snapshot pairs acquired with very different camera parameters, which could be a hint for detecting abrupt illumination changes. Thus, enriching the map with additional task-specific information has to be considered a means to improve our navigation strategies.

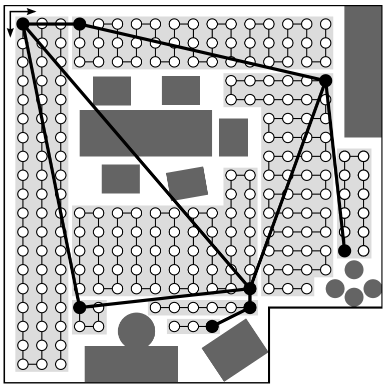


Figure 7.4.: Hierarchical mapping based on the decomposition of the robot's workspace into cleaning segments. Cleaning segments (light gray) are linked by the high-level, graph-based representation (black) if (i) place nodes of one segment were used for navigating along the first lane of the other cleaning segment or (ii) if loop closures between segments were detected. On the lower level, this information can be used to estimate metrical spatial interrelations between segments and to estimate the spatial positions of place nodes w.r.t. a common frame of reference (black arrows).

7.3.3. Hierarchical Mapping Based on the Decomposition Into Cleaning Segments

Based on the decomposition of the robot's workspace into several cleaning segments, interesting questions in the field of hierarchical mapping arise. Hierarchical maps consist of two levels of spatial representations (section 3.6.2): the *lower* level is formed by a set of local submaps each describing a region of the robot's workspace. The *higher* level represents spatial interrelations between submaps. In the context of our application (figure 7.4), the lower level corresponds to the cleaning segments obtained from either applying the trajectory controller proposed in chapter 4 or the particle-filter method by MÖLLER et al. [457]. Future work aims at finding the most appropriate hierarchical map for navigation of cleaning robots and to develop efficient algorithms operating on the maps.

For the higher-level representation, it is then straightforward to represent each segment by a graph node and to link neighboring segments. By this means, two segments are linked (i) if the border nodes of one segment were used for trajectory control along the first lane of the other segment or (ii) if loop closures between the two segments were detected. In the simplest case, the higher-level representation is a purely topological map and therefore does not contain metrical position information. However this would discard information, because a coarse estimate of the metrical relations between segments is available from the known extent of the segments. More accurate estimates could be derived from compass and home-direction estimates (i) already available from position estimation along the first lane or (ii) computed for image pairs identified by loop-closure detection. This suggests to also use a topo-metric map as higher-level representation. The open question regarding this issue is whether or not to correct the available position estimates stored in the high-level map if new information about the spatial interrelations of segments becomes available —e.g. when closing loops in the high-level map. On one side, we expect that the best results regarding mapping accuracy are obtained if the available estimates are updated based on the new information. This leads to a hierarchical SLAM method which maintains estimates of the spatial relations between a set of local cleaning segments. With this method, only the high-level representation is influenced by position updates, whereas position estimates in the low-level submaps are kept constant after adding the place nodes to the map. Thus, we would further stick to our concept of locally consistent submaps which do not undergo position corrections by SLAM techniques. On the other side, one can argue that mapping accuracy is not essential for our purpose and that a coarse estimate of the spatial interrelations between segments is sufficient. In this case, the computational effort for subsequent position optimizations could be avoided, and estimates would be computed solely from the information available at the point in time when the cleaning segment is started.

7.3.4. Omnidirectional Vision as Multi-Purpose Sensor

The experiments described in this thesis reveal that omnidirectional vision is an appropriate sensor modality for cleaning-robot control. Besides for omnidirectional visual navigation, it can also be applied for several other purposes such as (i) obstacle detection (section 7.3.4.1), (ii) tilt detection (section 7.3.4.2), or (iii) gesture recognition (section 7.3.4.3). We consider these four working directions to be most promising and therefore discuss them in the following. For issues (i) and (ii), visual information can replace dedicated sensors otherwise used for these purposes. By this means, the hardware costs of a potential product can be reduced.

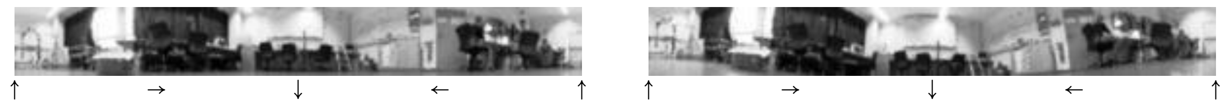
7.3.4.1. Visual Obstacle Detection

Obstacle detection based on omnidirectional image information can avoid hardware costs of distance sensors such as IR sensors or laser-range finders. Nevertheless, we expect that a bumper is still required as a backup sensor if other obstacle-detection methods fail. Visual obstacle detection can be categorized into *passive* and *active* approaches. *Passive approaches* solely rely on image information and detect obstacles from optical flow (textbooks: [305, 641], reviews: [30, 280]). Under translations of the robot, nearby obstacles are subject to a strong shift which allows to segment obstacles in the image. Since an estimate of the robot's motion between the two omnidirectional images can be derived either from odometry or from full pose estimation, we can compute the absolute distance of the detected obstacles, and the sensed obstacles can be integrated into a map. Optical flow-based obstacle detection allows for detecting obstacles in arbitrary viewing directions in the image. However, it requires structured image regions and cannot be applied for measuring the distance to large homogeneous regions such as walls. Currently, Andreas Stöckel develops such a method as a student project being supervised mainly by David Fleer and Michael Horst and to a lesser extent by Lorenz Hillen. Since the robot's motion between images is known, flow lines, i.e. the image regions along which obstacles move [216, 334], can be predicted. By restricting the search space for flow computations to these flow lines, the optical flow can be estimated accurately and efficiently. Hence, the method relies on similar principles than flow-line matching method for local visual homing developed by Lorenz Hillen ([217] and section 3.5.2.3). First results of the obstacle detection method look promising. Related approaches which operate on omnidirectional vision but do not exploit the flow lines are described by [418, 670, 713].

Active approaches to obstacle detection project a light pattern or light spots onto surrounding obstacles and detect this patterns in the image. Based on the position in the image, the distance to the obstacle can be computed. Such methods are also referred to as structured light (review: [179]). Their advantage is that they also allow for measuring the distance to obstacles with a homogeneous surface; their disadvantages include (i) to only provide sparse distance information into a small number of predefined directions and (ii) that additional light-emitting devices are required which also consume battery power. This line of research is investigated by Michael Horst [289]. The obstacle maps produced by visual distance measurement with a set of spot-laser diodes are of considerably better quality than the maps produced with the previously used Sharp GP2D12 and GP2D120 sensors [D12, D13]. Related approaches include [496, 712, 723] for omnidirectional vision and [201, 316, 317, 392] for directed cameras, respectively.

7.3.4.2. Visual Tilt Detection

Usually, cleaning robots are moving in the ground plane. In this case, vertical structures in the robot's environment are —if the navigation strategy operates on camera images without image unfolding— imaged as radial lines intersecting in the camera's principle point. If unfolded panoramas are used, vertical structures are imaged as vertical lines (section 3.2.4.1 and figure 3.8). However, in certain situations like moving over a carpet border or over other small steps, a tilt of the robot can occur. In this case, vertical lines in the world are no longer imaged as vertical lines (figure 7.5), but characteristic deviations occur. These deviations can be analyzed in order to estimate and later on to compensate the robot's tilt. As vertical structures in the world are usually visible as edges in



(1) Untilted robot. **(2)** Tilted robot.

Figure 7.5.: Image disturbances resulting from a tilt of the robot. In subfigure (1), the robot is not tilted. Vertical lines in the world are imaged as vertical lines and the horizon (in the shown image close to the lower image border) is imaged as horizontal line. Subfigure (2) shows a panoramic image if the robot is tilted for 6° into its forward direction. Only vertical lines in forward (↑) and backward (↓) direction are imaged as vertical lines; vertical lines in lateral viewing directions (←,→) appear slanted. Furthermore, the horizon is imaged as sinusoidal line. Used with kind permission of Martin Krzykawski.

the image, the tilt could be detected by applying steerable edge-detection filters. Steerable filters are rotation-sensitive filter kernels with Gabor filters being one of the most well known examples (textbooks: [186, 305]). By visual tilt detection, the costs for an inertial measurement unit (IMU) could be avoided. David Fleer recently started to investigate this research direction.

7.3.4.3. Gesture Recognition

Gesture recognition enables the user to easily communicate the robot what it is supposed to do. For example, a waving gesture could attract the robot's attention and could serve as a command to make the robot move towards the user. Another idea is to use pointing gestures to show the robot locations, which it is supposed to particularly clean (e.g. by a spot cleaning program). Related work on gesture recognition for robot navigation includes [98, 405, 573]. Beyond that, human robot interaction based on gestures is strongly influenced by sensors such as the Microsoft Kinect [I69] and the Asus Xtion PRO LIVE [I12], and it is an open question how approaches developed for these sensors can be applied for data obtained from omnidirectional cameras.

7.3.5. Summary and Appraisal of Future Work

The four main directions of future work outlined in sections 7.3.1 to 7.3.4 clearly aim towards a full-fledged cleaning robot. They are therefore an extension of the basic building blocks considered in this thesis. We expect that our future work (i) will bring us closer to a potential product and (ii) will also prove the feasibility and applicability of the building blocks and algorithms proposed in this thesis for a full-fledged cleaning robot. The work will certainly contribute to advance the field of mobile robots using omnidirectional vision as primary sensory information and to reach the goal of bringing such a robot into the customer's home. It was exciting to see the field grow and proceed during the years we worked on this dissertation, and it will also be exciting to see the developments until this goal is reached (and of cause also beyond).

A. Lane Controller With Partial Position Information

Table A.1.: Tabular presentation for results of true inter-lane distances as depicted in figure 4.17. Subtables (1) and (2) contain the results for experiments 1 and 2, respectively. Pooled percentiles were obtained by pooling over all lanes and are in section 4.5.2 referred to as $\bar{P}_x$. Abbreviations are as follows: Perc.: percentile; Pld.: pooled; all results in centimeters.

(1) Experiment 1 (data depicted in figure 4.17.1).

	Lane i							
Perc.	1	2	3	4	5	6	7	Pld.
$P_{0.05}^{(i)}$	26.9	22.4	21.3	21.6	19.8	21.4	19.0	20.9
$P_{0.25}^{(i)}$	30.6	26.6	26.8	25.3	25.8	25.9	24.4	26.3
$P_{0.50}^{(i)}$	32.9	29.1	30.3	28.9	28.8	29.7	27.9	29.8
$P_{0.75}^{(i)}$	35.2	31.7	32.8	31.8	30.6	32.5	30.8	32.4
$P_{0.95}^{(i)}$	38.5	37.0	38.4	36.1	33.7	37.3	34.7	37.0

(2) Experiment 2 (data depicted in figure 4.17.2).

	Lane i														
Perc.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	Pld.
$P_{0.05}^{(i)}$	23.9	21.3	28.3	21.1	25.8	20.5	21.7	18.7	22.3	15.0	18.3	16.1	21.8	16.4	19.4
$P_{0.25}^{(i)}$	29.1	26.3	30.8	26.9	29.7	24.7	27.8	23.8	26.4	19.8	24.5	26.2	28.3	24.9	26.2
$P_{0.50}^{(i)}$	30.5	29.0	34.2	29.7	32.0	27.9	29.9	26.7	29.9	24.2	30.1	30.1	30.9	29.4	29.8
$P_{0.75}^{(i)}$	32.7	30.7	37.9	32.2	35.8	30.1	32.7	29.8	32.2	28.9	31.8	32.7	33.0	31.5	32.2
$P_{0.95}^{(i)}$	37.4	34.6	42.2	37.3	39.4	32.3	37.4	32.0	35.4	32.2	35.1	40.1	36.7	39.2	37.7

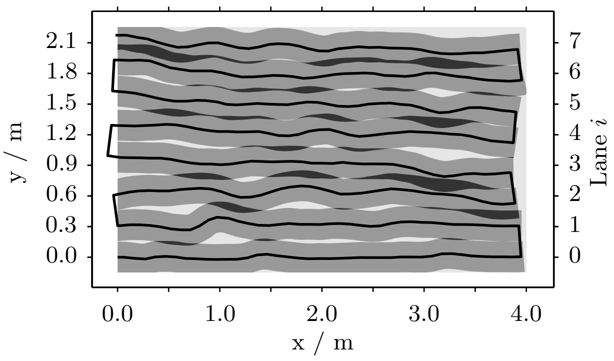
Table A.2.: Tabular presentation for results of cleaning performance as depicted in figure 4.18. Subtables (1) and (2) contain the results for experiments 1 and 2, respectively. All results in percent. A_0, A_1, and A_2 refer to the uncovered area, the area covered once, and the area covered repeatedly. The last column contains the average computed over all trials; these values are in section 4.5.3 referred to as $\bar{A}_0$, $\bar{A}_1$, and $\bar{A}_2$. Trials 1–5 were biased with a systematic error (section 4.4.4) causing the robot moving to the left ($k = -0.05$); for trials 6–10, the robot was biased to the right ($k = +0.05$). The highlighted columns mark the trials visualized in figures 4.18.1 and 4.18.2.

(1) Experiment 1

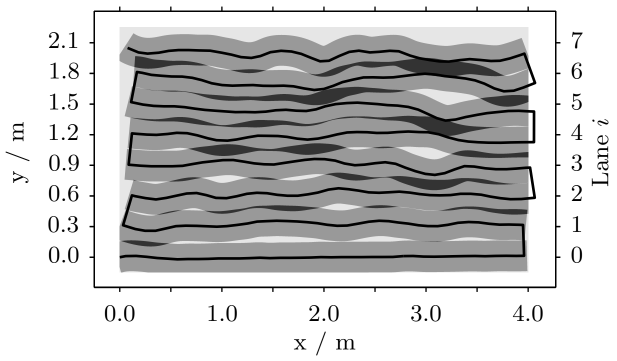
	Trial										
	1	2	3	4	5	6	7	8	9	10	Avg.
A_0	7.4	9.4	7.8	7.3	6.4	7.8	7.6	7.9	8.1	8.8	7.9
A_1	85.9	81.6	86.4	86.4	87.7	85.1	85.3	84.8	85.8	85.2	85.4
A_2	6.7	9.0	5.9	6.3	5.9	7.1	7.1	7.3	6.1	6.0	6.7

(2) Experiment 2

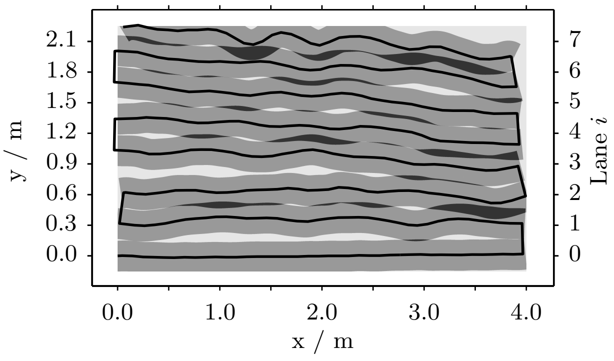
	Trial										
	1	2	3	4	5	6	7	8	9	10	Avg.
A_0	5.9	14.6	9.7	9.6	6.6	9.6	8.4	10.4	17.2	11.3	10.3
A_1	89.3	75.5	82.6	84.6	86.9	81.1	82.6	79.4	71.7	81.5	81.5
A_2	4.8	9.8	7.7	5.9	6.6	9.3	9.0	10.2	11.1	7.2	8.1



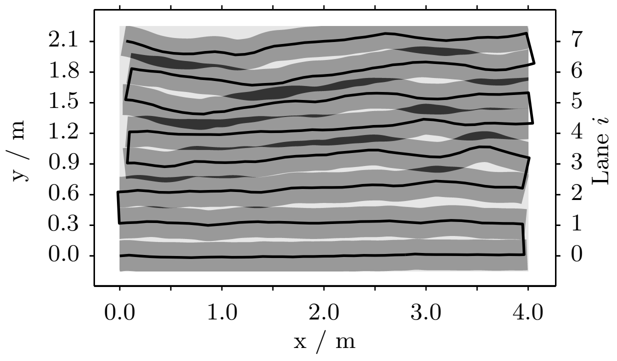
(1) Trial 1



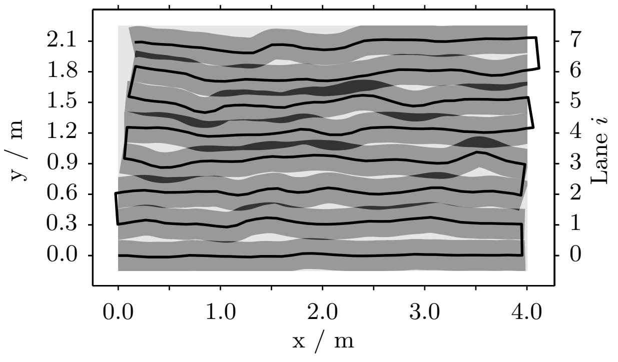
(2) Trial 2



(3) Trial 3

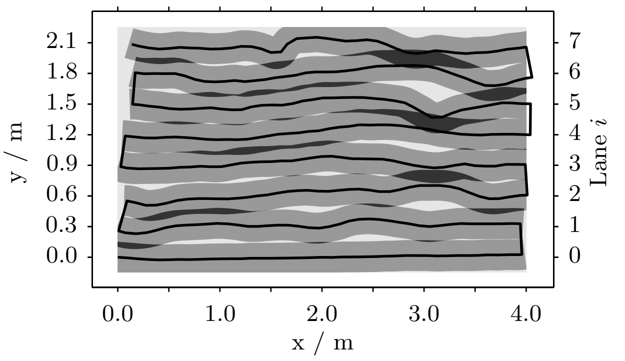


(4) Trial 4

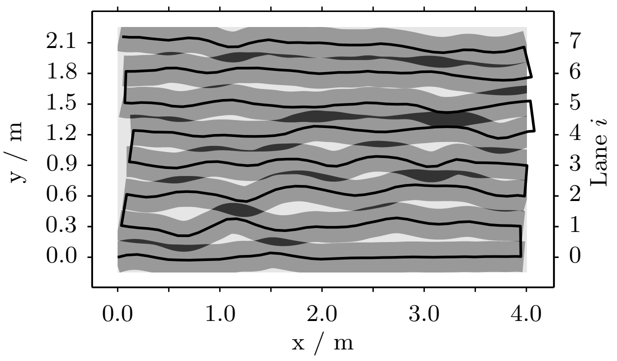


(5) Trial 5

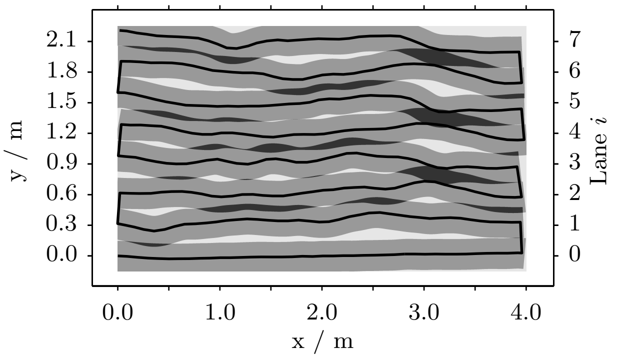
Figure A.1.: Cleaning performance of experiment 1. The subfigures (1) to (5) depict trials 1 to 5 of the experiment and were performed with a systematic error $k = -0.05$ causing the robot to turn to the left. Each plot shows the robot's trajectory (black line) together with the uncovered area A_0 (light gray), the area A_1 covered once (gray), and the area A_2 covered twice (dark gray).



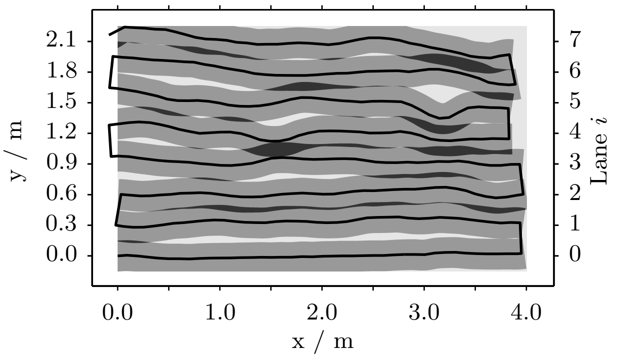
(1) Trial 6



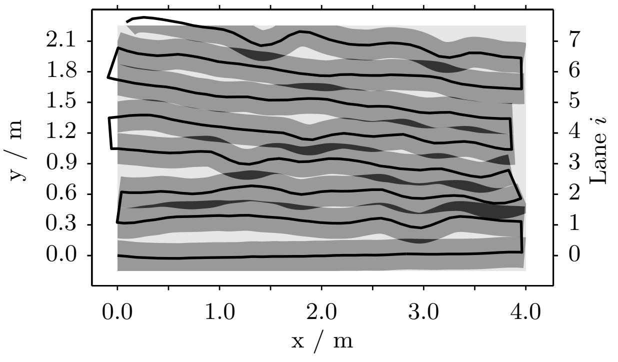
(2) Trial 7 (also depicted in figure 4.18.1)



(3) Trial 8

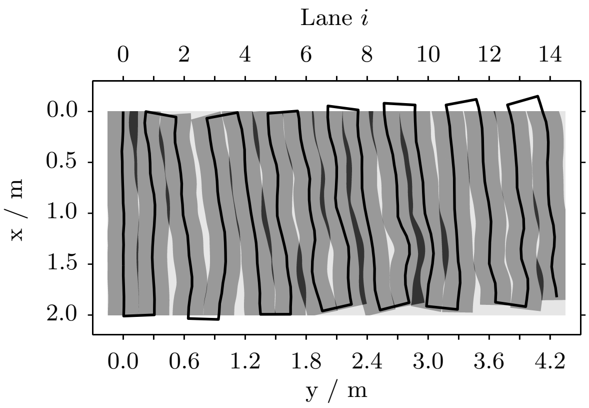


(4) Trial 9

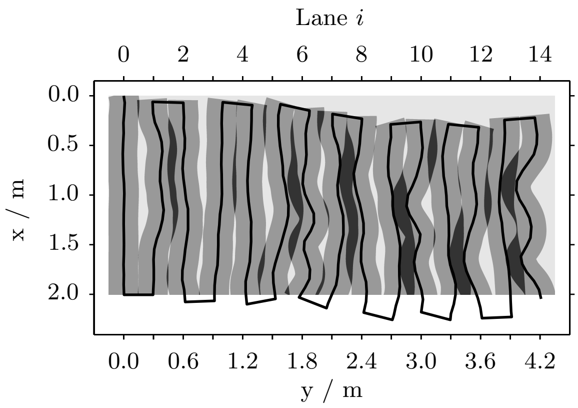


(5) Trial 10

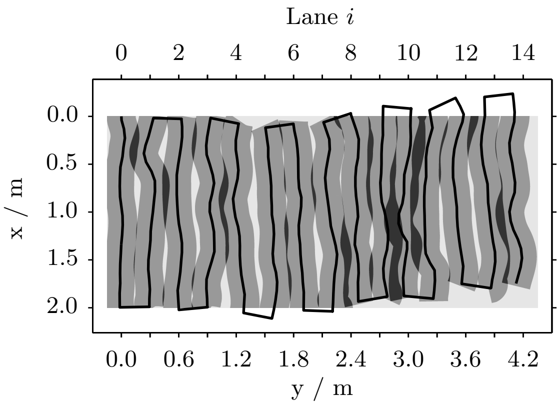
Figure A.2.: Cleaning performance of experiment 1. The subfigures (1) to (5) depict trials 6 to 10 of the experiment and were performed with a systematic error $k = 0.05$ causing the robot to turn to the right. Each plot shows the robot's trajectory (black line) together with the uncovered area A_0 (light gray), the area A_1 covered once (gray), and the area A_2 covered twice (dark gray).



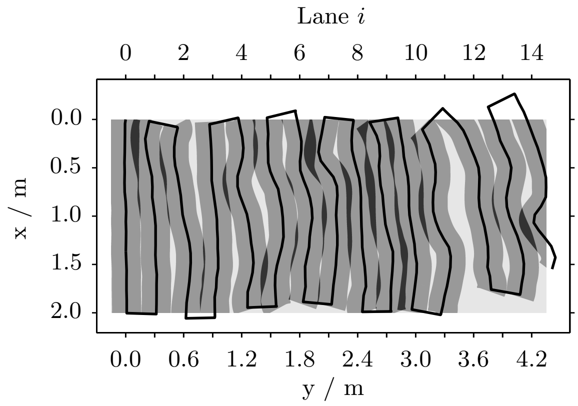
(1) Trial 1



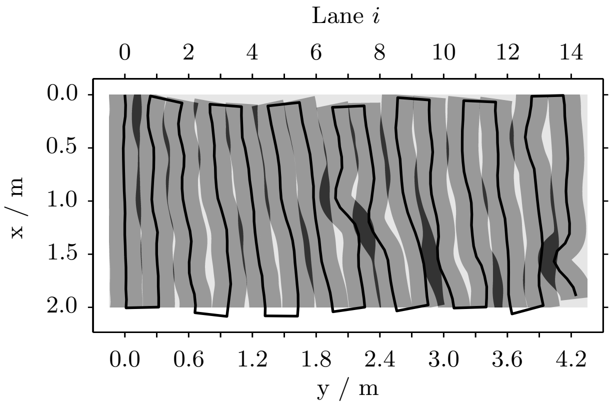
(2) Trial 2



(3) Trial 3

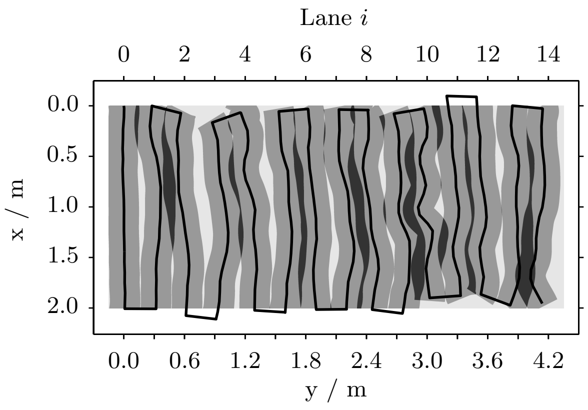


(4) Trial 4

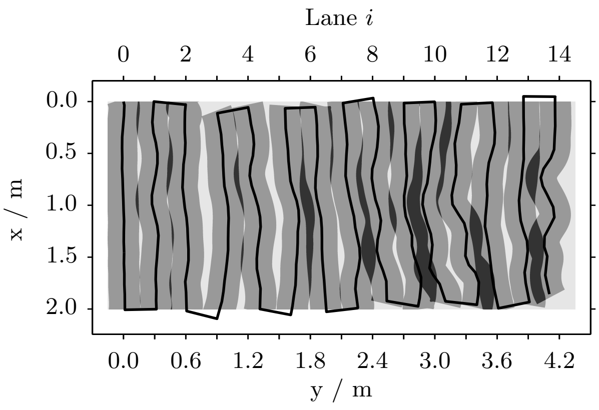


(5) Trial 5

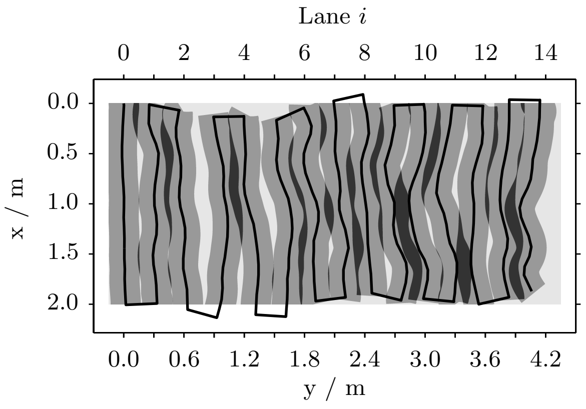
Figure A.3.: Cleaning performance of experiment 2. The subfigures (1) to (5) depict trials 1 to 5 of the experiment and were performed with a systematic error $k = -0.05$ causing the robot to turn to the left. Each plot shows the robot's trajectory (black line) together with the uncovered area A_0 (light gray), the area A_1 covered once (gray), and the area A_2 covered twice (dark gray).



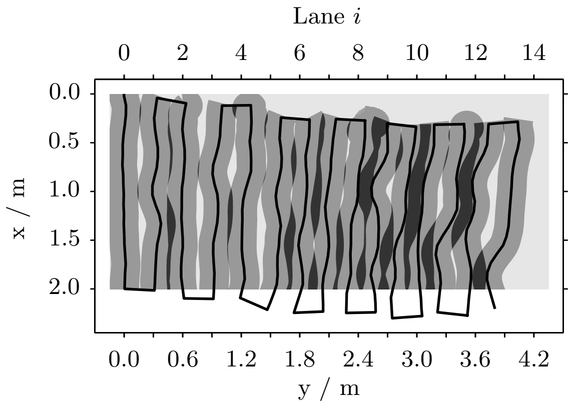
(1) Trial 6



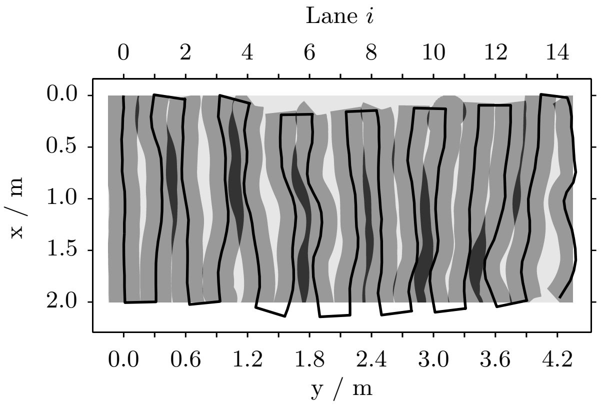
(2) Trial 7



(3) Trial 8



(4) Trial 9

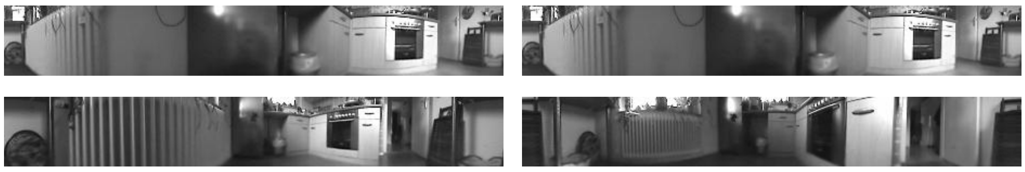


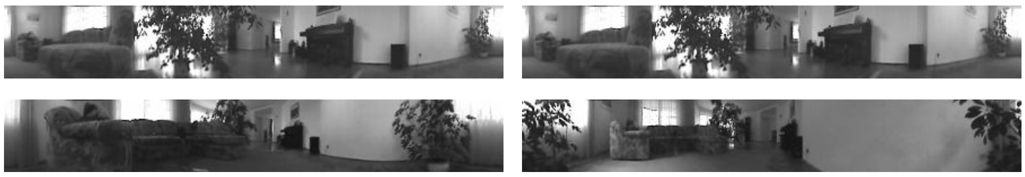
(5) Trial 10 (also depicted in figure 4.18.2)

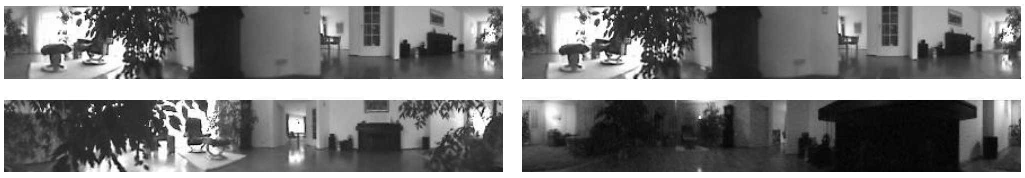
Figure A.4.: Cleaning performance of experiment 2. The subfigures (1) to (5) depict trials 6 to 10 of the experiment and were performed with a systematic error $k = 0.05$ causing the robot to turn to the right. Each plot shows the robot's trajectory (black line) together with the uncovered area A_0 (light gray), the area A_1 covered once (gray), and the area A_2 covered twice (dark gray).

B. Panoramic Image Databases

B.1. Example Images


(1) `kitchen`


(2) `moeller1`


(3) `moeller2`

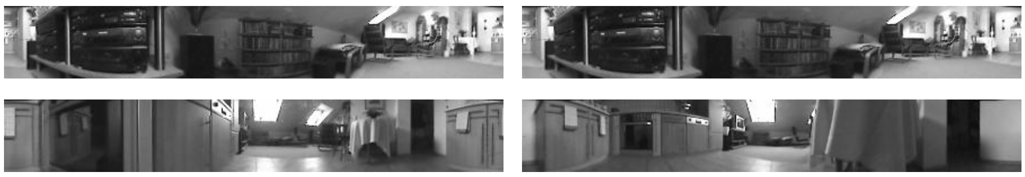
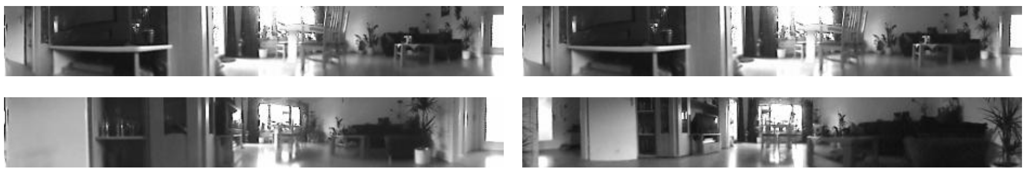

(4) `roeben`

Figure B.1.: Image databases of the `day|day` test group (figure 1 of 2; continued on the following page). For each of the databases subsumed in this test group (`kitchen`, `moeller1`, `moeller2`, `roeben`, `living1day`, `living2day`, `living3day`, and `living4day`), the images corresponding to the four corners of the database are shown in the subfigures. The image databases were collected in real apartments (the positions are visualized in figure B.4) under natural illumination conditions. Smaller databases were collected under (nearly) constant illumination conditions, larger databases are subject to moderate illumination changes. All databases were collected by Dr. Sven Kreft and Sebastian Ruwisch during their diploma theses; the work was supervised by Dr. Frank Röben and Lorenz Hillen. To achieve a high precision, the database-collection method proposed in the diploma thesis of Dr. Sven Kreft [342] was applied. It uses an additional camera observing a thread clamped on the floor. The thread is too thin to be visible in the panoramic database images and carries distance markers. By tracking the tread and the markers, the robot's position can be computed and its orientation is kept constant while moving along the thread. The method positions the robot with an accuracy of less than 1 cm; the orientation accuracy is better than 1°.

(1) `living1day`

(2) `living2day`


(3) `living3day`

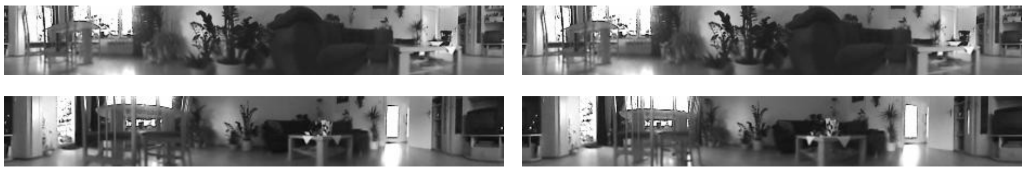
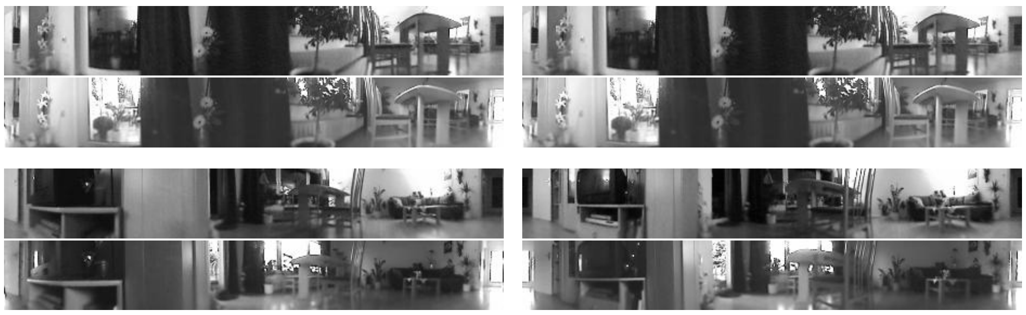
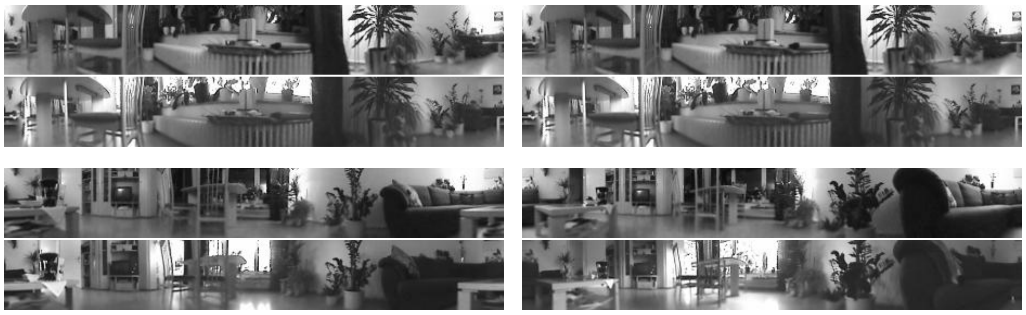

(4) `living4day`

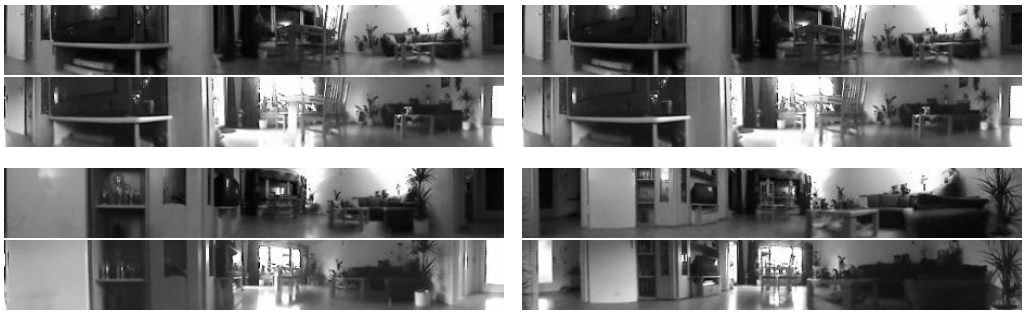
Figure B.2.: Image databases of the `day|day` test group (figure 2 of 2; continued from previous page).



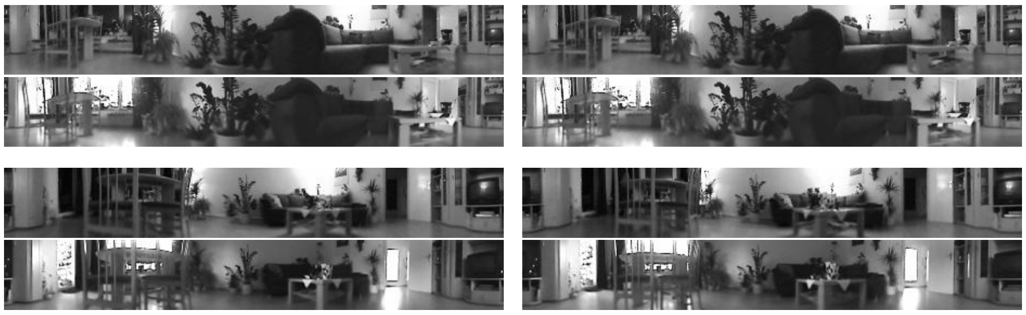
(1) `cliving1day`, `cliving1night`



(2) `cliving2day`, `cliving2night`



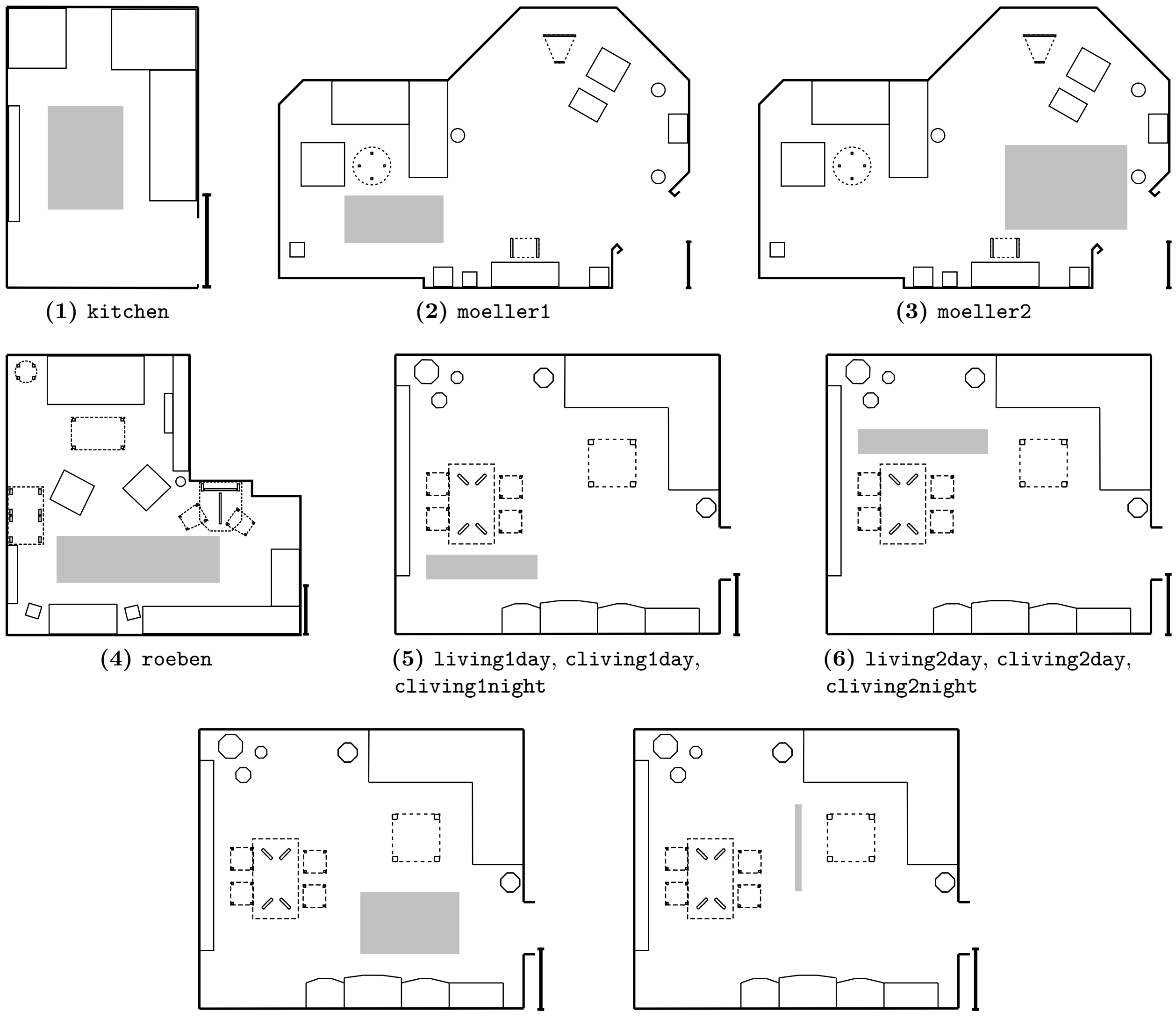
(3) `cliving3day`, `cliving3night`



(4) `cliving4day`, `cliving4night`

Figure B.3.: Image databases of the `day|night` test group. For each of the databases of this test group (`cliving1day`, `cliving1night`, `cliving2day`, `cliving2night`, `cliving3day`, `cliving3night`, `cliving4day`, and `cliving4night`), the images corresponding to the four corners of the database are shown in the subfigures. The lower image of each corner is the image acquired during day, the upper image was acquired during night. The databases were collected at identical positions once during day under natural illumination conditions and during night under constant artificial illumination conditions and allow for simulating changes of the illumination. The databases were acquired by Dr. Sven Kreft and Sebastian Ruwisch with the method briefly described in the caption of figure B.1; the locations of the databases in the apartments are depicted in figure B.4.

B.2. Positions of Image Acquisition



(1) `kitchen` (2) `moeller1` (3) `moeller2`

(4) `roeben` (5) `living1day`, `cliving1day`, `cliving1night` (6) `living2day`, `cliving2day`, `cliving2night`

(7) `living3day`, `cliving3day`, `cliving3night` (8) `living4day`, `cliving4day`, `cliving4night`

Figure B.4.: Spatial layout of apartments where image databases were collected. Each subfigure shows a single position of database acquisition; if several database were collected in an apartment, the databases are depicted in several subfigures. Visualized are the apartment's footprint (thick solid lines), ground-level furniture (thin solid lines), obstacles above ground (thin dashed lines), and the database area (gray filled rectangle). The measuring line at the right lower corner of each subfigure shows a length of 1 m.

C. Holistic Loop-Closure Detection and Visual Compass

C.1. Derivation of the Distance Measures Proposed by Möller [448, 449]

This section briefly recapitulates the derivation of the distance measures proposed in Möller [448, 449]. The distance measures are based on the Euclidean distance and were designed to reduce the influence of changes of the illumination modeled by a linear intensity transformation of the compared images (equation (5.52)).

The $\mathrm{d}_{\mathsf{asc}}$ function is derived to compensate for intensity scalings (Möller [449]). In order to obtain a symmetric measure, both images are scaled symmetrically:

$$\mathrm{d}_{\mathsf{asc}}(\boldsymbol{P},\boldsymbol{Q},a) = \frac{1}{2}\left\|\frac{1}{\sqrt{a}}\boldsymbol{P} - \sqrt{a}\boldsymbol{Q}\right\|^2 \tag{C.1}$$

Minimizing subject to a yields

$$a_{\min} = \frac{\|\boldsymbol{P}\|}{\|\boldsymbol{Q}\|}. \tag{C.2}$$

Inserting this result in equation (C.1) gives

$$\mathrm{d}_{\mathsf{asc}} = \|\boldsymbol{P}\|\|\boldsymbol{Q}\| - \langle\boldsymbol{P},\boldsymbol{Q}\rangle. \tag{C.3}$$

The dissimilarity function $\mathrm{d}_{\mathsf{aoc}}$ was proposed in Möller [448] and is supposed to compensate for constant shifts of the image intensities. The derivation minimizes

$$\mathrm{d}_{\mathsf{aoc}}(\boldsymbol{P},\boldsymbol{Q},o) = \frac{1}{2}\|\boldsymbol{P} - \boldsymbol{Q} + o\mathbf{1}\|^2 \tag{C.4}$$

subject to o resulting in

$$o_{\min} = \frac{1}{wh}(P - Q) \tag{C.5}$$

with

$$P = \sum_{x=0}^{w-1}\sum_{y=0}^{h-1} P(x,y) \text{ and} \tag{C.6}$$

$$Q = \sum_{x=0}^{w-1}\sum_{y=0}^{h-1} Q(x,y). \tag{C.7}$$

By combining these equations one obtains

$$\mathrm{d}_{\mathsf{aoc}} = \frac{1}{2}\left(\left(\|\boldsymbol{P}\|^2 - \frac{1}{wh}P^2\right) + \left(\|\boldsymbol{Q}\|^2 - \frac{1}{wh}Q^2\right)\right) - \langle\boldsymbol{P},\boldsymbol{Q}\rangle + \frac{1}{wh}PQ \tag{C.8}$$

$$= wh\left(\frac{\mathrm{var}(\boldsymbol{P}) + \mathrm{var}(\boldsymbol{Q})}{2} - \mathrm{cov}(\boldsymbol{P},\boldsymbol{Q})\right). \tag{C.9}$$

In order to compensate for scalings of the intensity a and constant shifts of the brightness o, the dissimilarity function $\mathrm{d}_{\mathtt{asoc}}$ was proposed in Möller [448]. For the derivation,

$$\mathrm{d}_{\mathtt{asoc}}(\boldsymbol{P}, \boldsymbol{Q}, a, o) = \frac{1}{2} \| \frac{1}{\sqrt{a}} \boldsymbol{P} - \sqrt{a} \boldsymbol{Q} + o\mathbf{1} \|^2 \tag{C.10}$$

is minimized with respect to a and o. This yields

$$a_{\min} = \sqrt{\frac{\|\boldsymbol{P}\|^2 - \frac{1}{wh} P^2}{\|\boldsymbol{Q}\|^2 - \frac{1}{wh} Q^2}} \text{ and} \tag{C.11}$$

$$o_{\min} = \frac{1}{wh} \left(\sqrt{a} \boldsymbol{Q} - \frac{1}{\sqrt{a} \boldsymbol{P}} \right). \tag{C.12}$$

Thus, we obtain

$$\mathrm{d}_{\mathtt{asoc}}(\boldsymbol{P}, \boldsymbol{Q}) = \sqrt{\left(\|\boldsymbol{P}\|^2 - \frac{1}{wh} P^2 \right) \left(\|\boldsymbol{Q}\|^2 - \frac{1}{wh} Q^2 \right)} - \langle \boldsymbol{P}, \boldsymbol{Q} \rangle + \frac{1}{wh} PQ \tag{C.13}$$

$$= wh \left(\sqrt{\operatorname{var}(\boldsymbol{P}) \operatorname{var}(\boldsymbol{Q})} - \operatorname{cov}(\boldsymbol{P}, \boldsymbol{Q}) \right). \tag{C.14}$$

C.2. Influence of Linearly Transformed Pixel Intensities on Preprocessed Images

Changes of the illumination conditions can be modelled by a linear transformation

$$\boldsymbol{I}' = a\boldsymbol{I} + o \tag{C.15}$$

of pixel intensities $\boldsymbol{I}$ (equation (5.52)). In this section, we analyze how a linear transformation of the input images $\boldsymbol{I}$ influences the images obtained from preprocessing the input images with the preprocessing functions described in section 5.2.1.

The preprocessing functions $\mathrm{p}_{\mathtt{pw}}$, $\mathrm{p}_{\mathtt{sob}}$, $\mathrm{p}_{\mathtt{lap}}$, and $\mathrm{p}_{\mathtt{dog}}$ (equations (5.8) to (5.11)) rely on a convolution of the image $\boldsymbol{I}$ with the kernel $\boldsymbol{K}_{\mathtt{x}}$.

Hence, a linear transformation of the input image yields

$$\mathrm{p}_{\mathtt{x}}(a\boldsymbol{I} + o) = \boldsymbol{K}_{\mathtt{x}} * (a\boldsymbol{I} + o) \tag{C.16}$$

$$= \boldsymbol{K}_{\mathtt{x}} * (a\boldsymbol{I}) + \boldsymbol{K}_{\mathtt{x}} * (o\mathbf{1}) \tag{C.17}$$

$$= a(\boldsymbol{K}_{\mathtt{x}} * \boldsymbol{I}) + o(\boldsymbol{K}_{\mathtt{x}} * \mathbf{1}) \tag{C.18}$$

where $\mathbf{1}$ is an image with constant pixel intensities of 1. With $\sum_{x,y} K_{\mathtt{x}}(x, y) = 0$, this results in

$$= a(\boldsymbol{K}_{\mathtt{x}} * \boldsymbol{I}) \tag{C.19}$$

$$= a\, \mathrm{p}_{\mathtt{x}}(\boldsymbol{I}) \tag{C.20}$$

Hence, the convolution with a kernel $\boldsymbol{K}_{\mathtt{x}}$ with entries summing up to 0 preserves the scale change a whereas the offset o is canceld out. In contrast to the second-order edge detectors $\mathrm{p}_{\mathtt{lap}}$ and $\mathrm{p}_{\mathtt{lap}}$, the first order edge detectors $\mathrm{p}_{\mathtt{pw}}$ and $\mathrm{p}_{\mathtt{sob}}$ filter the image with a horizontal and a vertical Kernel and combine the resulting images by computing the magnitude:

$$\begin{aligned}
\mathrm{p}_{\mathtt{x}}(a\boldsymbol{I}+o) &= \sqrt{(\boldsymbol{K}_{\mathtt{x}} * (a\boldsymbol{I}+o))^2 + (\boldsymbol{K}_{\mathtt{x}}^\top * (a\boldsymbol{I}+o))^2} && \text{(C.21)}\\
&= \sqrt{(a(\boldsymbol{K}_{\mathtt{x}} * \boldsymbol{I}))^2 + (a(\boldsymbol{K}_{\mathtt{x}}^\top * \boldsymbol{I}))^2} && \text{(C.22)}\\
&= a\sqrt{(\boldsymbol{K}_{\mathtt{x}} * \boldsymbol{I}))^2 + (\boldsymbol{K}_{\mathtt{x}}^\top * \boldsymbol{I})^2} && \text{(C.23)}\\
&= a\,\mathrm{p}_{\mathtt{x}}(\boldsymbol{I}) && \text{(C.24)}
\end{aligned}$$

Thus, computing the preprocessing functions $\mathrm{p}_{\mathtt{pw}}$ and $\mathrm{p}_{\mathtt{sob}}$ preserve the scale parameter a while the offset o is cancelled out.

The preprocessing functions $\mathrm{p}_{\mathtt{dl}}$ (equation (5.15)), $\mathrm{p}_{\mathtt{dv}}$ (equation (5.16)), and $\mathrm{p}_{\mathtt{dlc}}$ (equation (5.17)) inspired by early vision models rely on the local luminance L (equation (5.13)) and the variation of local luminance D (equation (5.13)). Therefore, we first analyze the robustness against changes of the illumination of these building blocks:

$$\begin{aligned}
\mathrm{L}(a\boldsymbol{I}+o) &= \boldsymbol{K}_{\mathtt{gau}}(\sigma) * (a\boldsymbol{I}+o) && \text{(C.25)}\\
&= \boldsymbol{K}_{\mathtt{gau}}(\sigma) * (a\boldsymbol{I}) + \boldsymbol{K}_{\mathtt{gau}}(\sigma) * (o\mathbf{1}) && \text{(C.26)}\\
&= a\boldsymbol{K}_{\mathtt{gau}}(\sigma) * (\boldsymbol{I}) + o && \text{(C.27)}\\
&= a\,\mathrm{L}(\boldsymbol{I}) + o \text{ (because } \sum_{x,y} a\boldsymbol{K}_{\mathtt{gau}}(\sigma) = 1) && \text{(C.28)}
\end{aligned}$$

and

$$\begin{aligned}
\mathrm{V}(a\boldsymbol{I}+o) &= \sqrt{\boldsymbol{K}_{\mathtt{gau}}(\sigma) * (a\boldsymbol{I}+o-\mathrm{L}(a\boldsymbol{I}+o))^2} && \text{(C.29)}\\
&= \sqrt{\boldsymbol{K}_{\mathtt{gau}}(\sigma) * (a\boldsymbol{I}+o-a\,\mathrm{L}(\boldsymbol{I})-o)^2} && \text{(C.30)}\\
&= a\sqrt{\boldsymbol{K}_{\mathtt{gau}}(\sigma) * (\boldsymbol{I}-\mathrm{L}(\boldsymbol{I}))^2} && \text{(C.31)}\\
&= \mathrm{V}(a\boldsymbol{I}+o). && \text{(C.32)}
\end{aligned}$$

With equations (C.28) and (C.32), one can analyze the influences of preprocessing an image with $\mathrm{p}_{\mathtt{dl}}$, $\mathrm{p}_{\mathtt{dv}}$, and $\mathrm{p}_{\mathtt{dl}}$, respectively:

$$\begin{aligned}
\mathrm{p}_{\mathtt{dl}}(a\boldsymbol{I}+o) &= \frac{\mathrm{V}(a\boldsymbol{I}+o)}{\mathrm{L}(a\boldsymbol{I}+o)} && \text{(C.33)}\\
&= \frac{a\,\mathrm{V}(\boldsymbol{I})}{a\,\mathrm{L}(\boldsymbol{I})+o} && \text{(C.34)}\\
&= \frac{\mathrm{V}(\boldsymbol{I})}{\mathrm{L}(\boldsymbol{I})+\frac{o}{a}} && \text{(C.35)}\\
&= \mathrm{p}_{\mathtt{dl}}\left(\boldsymbol{I}+\frac{o}{a}\right) && \text{(C.36)}
\end{aligned}$$

$$\begin{aligned}
\mathrm{p}_{\mathtt{dv}}(a\boldsymbol{I}+o) &= \frac{a\boldsymbol{I}+o-\mathrm{L}(a\boldsymbol{I}+o)}{1+\frac{\mathrm{V}(a\boldsymbol{I}+o)}{\sigma_{1/2}}} && \text{(C.37)}\\
&= \frac{a\boldsymbol{I}+o-a\,\mathrm{L}(\boldsymbol{I})-o}{1+\frac{a\,\mathrm{V}(\boldsymbol{I})}{\sigma_{1/2}}} && \text{(C.38)}\\
&= \mathrm{p}_{\mathtt{dv}}(a\boldsymbol{I}) && \text{(C.39)}
\end{aligned}$$

$$\mathrm{p}_{\mathtt{dlc}}(a\boldsymbol{I} + o) = \frac{a\boldsymbol{I} + o}{\mathrm{L}(a\boldsymbol{I} + o)\left(1 + \frac{\mathrm{V}(a\boldsymbol{I}+o)}{\mathrm{L}(a\boldsymbol{I}+o)}\right)} \tag{C.40}$$

$$= \frac{a\boldsymbol{I} + o}{a\,\mathrm{L}(\boldsymbol{I})\left(1 + \frac{a\,\mathrm{V}(\boldsymbol{I})}{a\,\mathrm{L}(\boldsymbol{I})+o}\right)} \tag{C.41}$$

$$= \frac{\boldsymbol{I} + \frac{o}{a}}{\mathrm{L}(\boldsymbol{I}) + \frac{o}{a} + \frac{\mathrm{V}(\boldsymbol{I})}{\mathrm{L}(\boldsymbol{I})+\frac{o}{a}}} \tag{C.42}$$

$$= \mathrm{p}_{\mathtt{dlc}}\left(\boldsymbol{I} + \frac{o}{a}\right). \tag{C.43}$$

From equations (C.36), (C.39) and (C.43) one can conclude that none of the preprocessing functions $\mathrm{p}_{\mathtt{dl}}$, $\mathrm{p}_{\mathtt{dv}}$, and $\mathrm{p}_{\mathtt{dlc}}$ can completely compensate for scalings a and shifts o of the image brightness. However, the preprocessing function $\mathrm{p}_{\mathtt{dv}}$ cancels out the offset o while preserving the scaling parameter a. The images resulting from preprocessing a linearly transformed input image with the functions $\mathrm{p}_{\mathtt{dl}}$ and $\mathrm{p}_{\mathtt{dlc}}$ are identical to preprocessing an image whose pixel intensities were not scaled but only shifted by $\frac{o}{a}$. Thus, arbitrary scalings a are compensated in case the image brightness is not shifted (i.e. $o = 0$).

C.3. Influence of Linearly Transformed Pixel Intensities on Image Dissimilarity Functions

In this section, we analyze how the dissimilarity values $\mathrm{d}(\boldsymbol{P}, a\boldsymbol{P} + o)$ obtained from various image dissimilarity functions d are influenced if one of the input images is linearly transformed according to equation (C.15).

For dissimilarity functions based on gray-value differences, it is not necessary to derive the complete mathematical terms for $\mathrm{d}_{\mathtt{x}}(\boldsymbol{P}, a\boldsymbol{P} + o)$. Rather, it is sufficient to analyze pixel differences d are computed; The image dissimilarity functions $\mathrm{d}_{\mathtt{sad}}$ (equation (5.22)), $\mathrm{d}_{\mathtt{maxn}}$ (equation (5.27)), $\mathrm{d}_{\mathtt{ssd}}$ (equation (5.28)), $\mathrm{d}_{\mathtt{eucl}}$ (equation (5.29)) and $\mathrm{d}_{\mathtt{rms}}$ (equation (5.30)) cannot compensate for linear transformations of the image brightness because none of the parameters a and o can be reduced. As it is impossible to draw the parameters out of the summations and the pixel differences, we expect these functions to be strongly influenced by changes of the illumination.

The zero-mean dissimilarity functions $\mathrm{d}_{\mathtt{zsad}}$ and $\mathrm{d}_{\mathtt{zssd}}$ compute pixel differences d after subtracting the average image brightness from each of the considered images $\boldsymbol{P}$ and $(a\boldsymbol{P} + o)$:

$$d = (P(x,y) - \bar{P}) - ((aP(x,y) + o) - (\overline{aP + o})) \tag{C.44}$$

$$= (P(x,y) - \bar{P}) - ((aP(x,y) + o) - (a\bar{P} + o)) \text{ (linearity of expectation)} \tag{C.45}$$

$$= (P(x,y) - \bar{P}) - a(P(x,y) - \bar{P}). \tag{C.46}$$

Thus, the distance functions $\mathrm{d}_{\mathtt{zsad}}$ and $\mathrm{d}_{\mathtt{zssd}}$ compensate for brightness shifts o but cannot compensate for changes of the illumination resulting in a scaling of the pixel intensities.

The dissimilarity functions $\mathrm{d}_{\mathtt{ssad}}$ and $\mathrm{d}_{\mathtt{sssd}}$ scale the pixel intensities of one input image with the ratio of the average image intensities before computing the pixel difference d:

$$d = P(x,y) - \frac{\bar{P}}{\overline{aP + o}}(aP(x,y) + o) \tag{C.47}$$

$$d = P(x,y) - \frac{\bar{P}}{a\bar{P} + o}(aP(x,y) + o) \tag{C.48}$$

$$d = P(x,y) - \frac{aP(x,y) + o}{a + \frac{o}{\bar{P}}}. \tag{C.49}$$

To this end, these dissimilarity functions do not compensate for a and o. However, in case the image brightness is not shifted (i.e. $o = 0$), they compensate for arbitrary scalings a.

The distance measures proposed by MÖLLER [448, 449] were derived to be invariant under brightness scalings ($\mathrm{d}_{\mathtt{asc}}$, equation (5.33)), under brightness shifts ($\mathrm{d}_{\mathtt{aoc}}$, equation (5.33)), and both scalings and shifts ($\mathrm{d}_{\mathtt{asc}}$, equation (5.33))). Thus, we find (for sake of simplicity we apply the notation of norm functions $\|.\|$ and the dot product $\langle .,. \rangle$ to images because an image can be simply transformed into a vector by stacking the image column-by-column):

$$\mathrm{d}_{\mathtt{asc}}(\boldsymbol{P}, a\boldsymbol{P} + o) = \|\boldsymbol{P}\|\|a\boldsymbol{P} + o\| - \langle \boldsymbol{P}, a\boldsymbol{P} + o\rangle \tag{C.50}$$

$$= \|\boldsymbol{P}\|\sqrt{\langle a\boldsymbol{P} + o, a\boldsymbol{P} + o\rangle} - \langle \boldsymbol{P}, a\boldsymbol{P} + o\rangle \tag{C.51}$$

$$= \|\boldsymbol{P}\|\sqrt{a^2\langle \boldsymbol{P}, \boldsymbol{P}\rangle + 2ao\langle \boldsymbol{P}, \mathbf{1}\rangle + o^2\langle \mathbf{1}, \mathbf{1}\rangle} - a\langle \boldsymbol{P}, \boldsymbol{P}\rangle - o\langle \boldsymbol{P}, \mathbf{1}\rangle \tag{C.52}$$

$$= \|\boldsymbol{P}\|\sqrt{a^2\langle \boldsymbol{P}, \boldsymbol{P}\rangle + 2ao\sum_{x=0}^{w-1}\sum_{y=0}^{h-1} P(x,y) + who^2} - a\langle \boldsymbol{P}, \boldsymbol{P}\rangle - o\sum_{x=0}^{w-1}\sum_{y=0}^{h-1} P(x,y) \tag{C.53}$$

and with assuming $o = 0$:

$$= a\|\boldsymbol{P}\|^2 - a\|\boldsymbol{P}\|^2 \tag{C.54}$$

$$= \mathrm{d}_{\mathtt{asc}}(\boldsymbol{P}, \boldsymbol{P}) \tag{C.55}$$

$$\mathrm{d}_{\mathtt{aoc}}(\boldsymbol{P}, a\boldsymbol{P} + o) = n\left(\frac{\mathrm{var}(\boldsymbol{P}) + \mathrm{var}(a\boldsymbol{P} + o)}{2} - \mathrm{cov}(\boldsymbol{P}, a\boldsymbol{P} + o)\right) \tag{C.56}$$

$$= n\left(\frac{\mathrm{var}(\boldsymbol{P}) + a^2\,\mathrm{var}(\boldsymbol{P})}{2} - a\,\mathrm{cov}(\boldsymbol{P}, \boldsymbol{P})\right) \tag{C.57}$$

$$= \mathrm{d}(\boldsymbol{P}, a\boldsymbol{P}) \tag{C.58}$$

$$\mathrm{d}_{\mathtt{asoc}}(\boldsymbol{P}, a\boldsymbol{P} + o) = n\left(\sqrt{\mathrm{var}(\boldsymbol{P})\,\mathrm{var}(a\boldsymbol{P} + o)} - \mathrm{cov}(\boldsymbol{P}, a\boldsymbol{P} + o)\right) \tag{C.59}$$

$$= n\left(\sqrt{a^2\,\mathrm{var}(\boldsymbol{P})^2} - a\,\mathrm{cov}(\boldsymbol{P}, \boldsymbol{P})\right) \tag{C.60}$$

$$= n\left(a\,\mathrm{var}(\boldsymbol{P}) - a\,\mathrm{var}(\boldsymbol{P})\right) \tag{C.61}$$

$$= \mathrm{d}_{\mathtt{asoc}}(\boldsymbol{P}, \boldsymbol{P}). \tag{C.62}$$

For the correlation-based dissimilarity measures $\mathrm{d}_{\mathtt{cc}}$ (equation (5.36)) and $\mathrm{d}_{\mathtt{ncc}}$ (equation (5.37)), the parameters a or o cannot be reduced. Due to the involved normalization, we expect the normalized cross-correlation $\mathrm{d}_{\texttt{ncc}}$ to be more robust against changes of the illumination than the standard cross-correlation coefficient $\mathrm{d}_{\texttt{cc}}$. The normalized cross-correlation $\mathrm{d}_{\texttt{zncc}}$ (equation (5.39)) completely compensates both scalings a and shifts o of the image intensities:

$$\begin{aligned}
\mathrm{d}_{\texttt{zncc}}(\boldsymbol{P}, a\boldsymbol{P}+o) &= \mathrm{d}_{\texttt{ncc}}\left(\boldsymbol{P}-\bar{P}, a\boldsymbol{P}+o-(\overline{aP(x,y)+o})\right) && \text{(C.63)}\\
&= \mathrm{d}_{\texttt{ncc}}\left(\boldsymbol{P}-\bar{P}, a\boldsymbol{P}+o-(a\bar{P}+o)\right) && \text{(C.64)}\\
&= \mathrm{d}_{\texttt{ncc}}\left(\boldsymbol{P}-\bar{P}, a(\boldsymbol{P}-\bar{P})\right) && \text{(C.65)}\\
&= \frac{\sum_{x=0}^{w-1}\sum_{y=0}^{h-1}(P(x,y)-\bar{P})(a(P(x,y)-\bar{P}))}{\sqrt{\sum_{x=0}^{w-1}\sum_{y=0}^{h-1}(P(x,y)-\bar{P})^2\sum_{x=0}^{w-1}\sum_{y=0}^{h-1}(a(P(x,y)-\bar{P}))^2}} && \text{(C.66)}\\
&= \frac{a\sum_{x=0}^{w-1}\sum_{y=0}^{h-1}(P(x,y)-\bar{P})(P(x,y)-\bar{P})}{a\sqrt{\sum_{x=0}^{w-1}\sum_{y=0}^{h-1}(P(x,y)-\bar{P})^2\sum_{x=0}^{w-1}\sum_{y=0}^{h-1}(P(x,y)-\bar{P})^2}} && \text{(C.67)}\\
&= \mathrm{d}_{\texttt{ncc}}\left(\boldsymbol{P}-\bar{P}, \boldsymbol{P}-\bar{P}\right) && \text{(C.68)}\\
&= \mathrm{d}_{\texttt{zncc}}(\boldsymbol{P}, \boldsymbol{P}) && \text{(C.69)}
\end{aligned}$$

C.4. Further Results

Table C.1.: Computing times for image preprocessing functions. The results are grouped into subfigures according to the taxonomy given in figure 5.3 and to the function's parameters.

(1) Edge detectors $\mathtt{p_{pw}}$, $\mathtt{p_{sob}}$ and $\mathtt{p_{lap}}$.

	$\mathtt{p_{pw}}$	$\mathtt{p_{sob}}$	$\mathtt{p_{lap}}$
Time / ms	6.91	6.91	0.38

(2) Difference of Gaussians $\mathtt{p_{dog}}$.

	$\mathtt{p_{dog1}}$	$\mathtt{p_{dog2}}$	$\mathtt{p_{dog3}}$	$\mathtt{p_{dog4}}$
σ_1	0.010	0.010	0.010	0.025
σ_2	0.020	0.040	0.080	0.050
Time / ms	8.95	8.94	9.76	9.09

(3) Local contrast $\mathtt{p_{dl}}$.

	$\mathtt{p_{dl1}}$	$\mathtt{p_{dl2}}$	$\mathtt{p_{dl3}}$	$\mathtt{p_{dl4}}$
w'	5	7	11	15
Time / ms	20.76	40.17	99.05	178.37

(4) Dividing by variation $\mathtt{p_{dlc}}$.

	$\mathtt{p_{dlc1}}$	$\mathtt{p_{dlc2}}$	$\mathtt{p_{dlc3}}$	$\mathtt{p_{dlc4}}$
w'	5	7	11	15
Time / ms	22.69	43.46	106.77	194.14

(5) Dividing by luminance/contrast $\mathtt{p_{dv}}$.

		$\mathtt{p_{dv*/1}}$	$\mathtt{p_{dv*/2}}$	$\mathtt{p_{dv*/3}}$	$\mathtt{p_{dv*/4}}$
		$w' = 5$	$w' = 7$	$w' = 11$	$w' = 15$
$\mathtt{p_{dv1/*}}$	$\sigma_{1/2} = 0.0$	22.61	43.17	106.32	192.71
$\mathtt{p_{dv2/*}}$	$\sigma_{1/2} = 0.5$	22.65	43.24	106.63	194.68
$\mathtt{p_{dv3/*}}$	$\sigma_{1/2} = 5.0$	22.68	43.38	106.10	194.38
$\mathtt{p_{dv4/*}}$	$\sigma_{1/2} = 50.0$	22.66	43.37	106.69	194.11

(6) Histogram equalization $\mathtt{p_{heq}}$.

	$\mathtt{p_{heq1}}$	$\mathtt{p_{heq2}}$	$\mathtt{p_{heq3}}$	$\mathtt{p_{heq4}}$	$\mathtt{p_{heq5}}$
b	4	8	16	32	64
Time / ms	1.57	1.60	1.60	1.59	1.59

Table C.2.: Computing times for global image comparison functions. The results are grouped into subfigures according to the taxonomy proposed in figure 5.4.

(1) Image dissimilarity functions relying on absolute pixel differences

	$\mathrm{d}_{\mathtt{sad}}$	$\mathrm{d}_{\mathtt{zsad}}$	$\mathrm{d}_{\mathtt{ssad}}$
Time / ms	24.35	31.45	27.68

(2) Image dissimilarity functions relying on squared pixel differences

	$\mathrm{d}_{\mathtt{ssd}}$	$\mathrm{d}_{\mathtt{eucl}}$	$\mathrm{d}_{\mathtt{rms}}$	$\mathrm{d}_{\mathtt{zssd}}$	$\mathrm{d}_{\mathtt{sssd}}$
Time / ms	24.15	24.23	24.21	32.37	27.65

(3) Alternative image dissimilarity functions relying on squared pixel differences

	$\mathrm{d}_{\mathtt{asc}}$	$\mathrm{d}_{\mathtt{aoc}}$	$\mathrm{d}_{\mathtt{asoc}}$
Time / ms	23.19	23.06	23.19

(4) Correlation measures

	$\mathrm{d}_{\mathtt{cc}}$	$\mathrm{d}_{\mathtt{ncc}}$	$\mathrm{d}_{\mathtt{zncc}}$
Time / ms	24.21	27.73	38.31

(5) Mutual information

	$\mathrm{d}_{\mathtt{mi1}}$	$\mathrm{d}_{\mathtt{mi2}}$	$\mathrm{d}_{\mathtt{mi3}}$	$\mathrm{d}_{\mathtt{mi4}}$	$\mathrm{d}_{\mathtt{mi5}}$
b	4	8	16	32	64
Time / ms	632.71	632.99	635.13	641.60	659.44

Table C.3.: AUC analysis for accelerated compass method. The ranking was obtained by sorting w.r.t. the AUC values. In case of identical AUC values, the overall number of coefficients $r \cdot b$ and the number of rings r were used as secondary and tertiary sorting criterion. Subfigures (1) and (2) show the results of the `day|day` and the `day|night` test groups, respectively.

(1) `day|day` group

p	r	b	$r \cdot b$	AUC
$\mathrm{p}_{\mathtt{dl4}}$	1	4	4	0.580
$\mathrm{p}_{\mathtt{dog4}}$	1	4	4	0.466
$\mathrm{p}_{\mathtt{dv4/4}}$	1	4	4	0.455
$\mathrm{p}_{\mathtt{sob}}$	1	4	4	0.401
$\mathrm{p}_{\mathtt{pw}}$	1	4	4	0.400
$\mathrm{p}_{\mathtt{dlc}}$	1	4	4	0.396
$\mathrm{p}_{\mathtt{lap}}$	1	4	4	0.395
$\mathrm{p}_{\mathtt{id}}$	1	4	4	0.390
$\mathrm{p}_{\mathtt{heq4}}$	1	4	4	0.373

(2) `day|night` group

p	r	b	$r \cdot b$	AUC
$\mathrm{p}_{\mathtt{dv1/4}}$	1	8	8	0.574
$\mathrm{p}_{\mathtt{heq4}}$	1	4	4	0.541
$\mathrm{p}_{\mathtt{lap}}$	2	230	460	0.535
$\mathrm{p}_{\mathtt{dlc1}}$	1	4	4	0.529
$\mathrm{p}_{\mathtt{dog4}}$	1	8	8	0.523
$\mathrm{p}_{\mathtt{dl4}}$	1	4	4	0.510
$\mathrm{p}_{\mathtt{id}}$	1	16	16	0.498
$\mathrm{p}_{\mathtt{pw}}$	1	8	8	0.456
$\mathrm{p}_{\mathtt{sob}}$	1	8	8	0.455

D. Signature-Based Loop-Closure Detection

D.1. Ground Distance and Relation to L_1-Norm

Let $\boldsymbol{h}$ be an histogram with n equally spaced bins numbered from 0 to $n-1$. Furthermore, let b be the difference between two histogram borders of $\boldsymbol{h}$. Then, the representative r_i of bin i is given by

$$r_i = \frac{i}{n-1} + \frac{1}{2}b. \tag{D.1}$$

The L_1-norm between two bin representatives r_i and r_j is given by:

$$\begin{aligned}
\mathrm{d}(r_i, r_j) &= |r_i, r_j| && \text{(D.2)}\\
&= |\frac{i}{n-1} + \frac{1}{2}b - \frac{j}{n-1}i\frac{1}{2}b| && \text{(D.3)}\\
&= |\frac{i}{n-1} - \frac{i}{n-1}| && \text{(D.4)}\\
&= \frac{1}{n-1}|i-j| && \text{(D.5)}\\
&= \frac{1}{n-1}\,\mathrm{d}(i,j). && \text{(D.6)}
\end{aligned}$$

Thus, the L_1-norm between bin representatives r_i and r_j is identical up to scale to the absolute difference between bin numbers i and j. As all tested ground distances further normalize the resulting distance values by the largest possible distance value, the two distances become identical.

D.2. Influence of Linearly Transformed Pixel Intensities on Signature Functions

For parameter-based loop-closure detection, we aim at using signature functions, which are robust against changes of the illumination. Therefore, we analyze how a linear intensity transformation of the input image influences the resulting parameter vectors:

$$\boldsymbol{p}' = \mathrm{s}(a\boldsymbol{I} + o). \tag{D.7}$$

The signature functions $\mathrm{s}_{\texttt{hist}}$ and $\mathrm{s}_{\texttt{chist}}$ (equations (6.4) and (6.5)) are an estimator for the probability density function and the cumulative density function, respectively, of image intensities. As a transformation of pixel intensities also changes the underlying distribution of gray values, the `hist`- and `chist`-signatures will also change. The influence of such an intensity transformation on the resulting histogram or cumulative histogram strongly depends on the number of histogram bins o. For small illumination changes (i.e. $a \approx 1$ and $o \approx 0$) and a small number of bins o, it is likely that the resulting histograms are approximately identical:

$$\begin{aligned}
\mathrm{s}_{\texttt{hist}}(a\boldsymbol{I} + o) &\approx \mathrm{s}_{\texttt{hist}}(\boldsymbol{I}) && \text{(D.8)}\\
\mathrm{s}_{\texttt{chist}}(a\boldsymbol{I} + o) &\approx \mathrm{s}_{\texttt{chist}}(\boldsymbol{I}). && \text{(D.9)}
\end{aligned}$$

The statistical signature functions combine the empirical moments to parameter vectors. We therefore analyse the influence of a linear intensity transformation onto the mean, variance, skewness, and kurtosis of image intensities:

$$\operatorname{mean}(a\boldsymbol{I} + o) = a \operatorname{mean}(\boldsymbol{I}) + o \tag{D.10}$$

$$\operatorname{var}(a\boldsymbol{I} + o) = a^2 \operatorname{var}(\boldsymbol{I}) \tag{D.11}$$

$$\begin{aligned}
\operatorname{skew}(a\boldsymbol{I} + o) &= \frac{1}{wh \cdot \operatorname{var}(a\boldsymbol{I} + o)^{\frac{3}{2}}} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \left((a\boldsymbol{I} + o)(x,y) - \operatorname{mean}(a\boldsymbol{I} + o)\right)^3 && \text{(D.12)}\\
&= \frac{1}{wh \cdot (a^2 \operatorname{var}(\boldsymbol{I}))^{\frac{3}{2}}} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \left(aI(x,y) + o - a \operatorname{mean}(\boldsymbol{I}) - o\right)^3 && \text{(D.13)}\\
&= \frac{1}{wh \cdot \operatorname{var}(\boldsymbol{I})^{\frac{3}{2}}} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \left(I(x,y) - \operatorname{mean}(\boldsymbol{I})\right)^3 && \text{(D.14)}\\
&= \operatorname{skew}(\boldsymbol{I}) && \text{(D.15)}
\end{aligned}$$

$$\begin{aligned}
\operatorname{kurt}(a\boldsymbol{I} + o) &= \frac{1}{wh \cdot \operatorname{var}(a\boldsymbol{I} + o)^2 - 3} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \left((a\boldsymbol{I} + o)(x,y) - \operatorname{mean}(a\boldsymbol{I} + o)\right)^4 && \text{(D.16)}\\
&= \frac{1}{wh \cdot a^4 \operatorname{var}(\boldsymbol{I})^2 - 3} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \left(aI(x,y) + o - a \operatorname{mean}(\boldsymbol{I}) - o\right)^4 && \text{(D.17)}\\
&= \frac{a^4}{wh \cdot a^4 \operatorname{var}(\boldsymbol{I})^2 - 3} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \left(I(x,y) - \operatorname{mean}(\boldsymbol{I})\right)^4 && \text{(D.18)}\\
&\approx \operatorname{kurt}(\boldsymbol{I}). && \text{(D.19)}
\end{aligned}$$

To this end, the signature functions $\mathrm{s}_{\texttt{ms}}$, $\mathrm{s}_{\texttt{mk}}$, and $\mathrm{s}_{\texttt{msk}}$ (equations (6.8), (6.9) and (6.11)) completely compensate for linear intensity transformations. The functions $\mathrm{s}_{\texttt{mv}}$ and $\mathrm{s}_{\texttt{mvsk}}$ (equations (6.7) and (6.13)) compensate for brightness shifts whereas the functions $\mathrm{s}_{\texttt{mm}}$, $\mathrm{s}_{\texttt{mmv}}$, $\mathrm{s}_{\texttt{mmvs}}$ and $\mathrm{s}_{\texttt{mmvsk}}$ (equations (6.6), (6.10), (6.12) and (6.14)) incorporating the average image intensity $\operatorname{mean}(\boldsymbol{I})$ cannot compensate for linear transformation of the image intensities.

The signature function $\mathrm{s}_{\texttt{cog}}$ can only compensate for linear intensity transformations in special cases:

$$\begin{aligned}
\mathrm{s}_{\texttt{cog}}(a\boldsymbol{I} + o) &= \frac{1}{n'} \left\| \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} (a\boldsymbol{I} + o)(x,y) \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} \right\| && \text{(D.20)}\\
&= \frac{1}{n'} \left\| \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} (aI(x,y) + o) \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} \right\| && \text{(D.21)}
\end{aligned}$$

$$= \frac{1}{n'} \left\| a \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y) \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} + \underbrace{o \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix}}_{:=\boldsymbol{z}} \right\| \tag{D.22}$$

$$= \frac{a}{n'} \left\| \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y) \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} \right\| \tag{D.23}$$

$$= \frac{a}{\sum_{x=0}^{w-1} \sum_{y=0}^{h-1} (a\boldsymbol{I} + o)(x,y)} \left\| \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y) \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} \right\| \tag{D.24}$$

$$= \frac{a}{a \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y) + who} \left\| \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y) \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} \right\| \tag{D.25}$$

because

$$\boldsymbol{z} = o \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} \tag{D.26}$$

$$= o \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} y \begin{pmatrix} \cos\left(\frac{2\pi x}{w}\right) \\ \sin\left(\frac{2\pi x}{w}\right) \end{pmatrix} \tag{D.27}$$

$$= o \sum_{y=0}^{h-1} y \underbrace{\sum_{x=0}^{w-1} \begin{pmatrix} \cos\left(\frac{2\pi x}{w}\right) \\ \sin\left(\frac{2\pi x}{w}\right) \end{pmatrix}}_{=\boldsymbol{0}} \tag{D.28}$$

$$= \boldsymbol{0}. \tag{D.29}$$

To this end, the signature function $\mathrm{s}_{\mathsf{cog}}$ cannot compensate for linear intensity changes. However, in case the average image intensity does not change (i.e. $o = 0$), arbitrary scale changes are compensated for. In this case, equation (D.25) becomes

$$= \frac{1}{\sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y)} \left\| \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} I(x,y) \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} \right\| \tag{D.30}$$

$$= \mathrm{s}_{\mathsf{cog}}(\boldsymbol{I}). \tag{D.31}$$

A one-dimensional row-image $\boldsymbol{I}(j)$ with pixels $0 \le j < w$ can be written as Fourier series (e.g. [611], textbook: [65])

$$I(\phi) = \frac{1}{2} a_0 + \sum_{k=1}^{\infty} \left(\mathfrak{a}_k \cos(k\phi) + \mathfrak{b}_k \sin(k_\phi) \right) \tag{D.32}$$

with coefficients

$$\mathfrak{a}_k = \frac{2}{w} \sum_j I(j) \cos\left(k \frac{2\pi}{w} j\right) \text{ and} \tag{D.33}$$

$$\mathfrak{b}_k = \frac{2}{w} \sum_j I(j) \sin\left(k \frac{2\pi}{w} j\right). \tag{D.34}$$

For $k = 0$ the component

$$\mathfrak{a}_0 = \frac{2}{w}\sum_j I(j)\cos(0\frac{2\pi}{w}j) \tag{D.35}$$

$$= \frac{2}{w}\sum_j I(j) \tag{D.36}$$

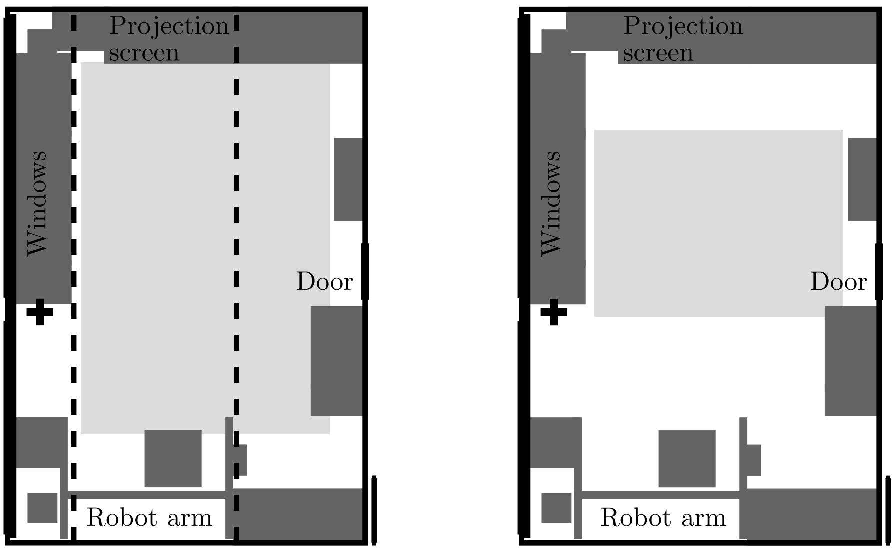
represents twice the average image intensity, whereas $\mathfrak{b}_0 = 0$. Larger coefficients therefore express the deviation from the average image intensity. By analyzing

$$\mathfrak{a}_k = \frac{2}{w}\sum_j (aI(j)+o)\cos\left(k\frac{2\pi}{w}j\right) \text{ and} \tag{D.37}$$

$$\mathfrak{b}_k = \frac{2}{w}\sum_j (aI(j)+o)\sin\left(k\frac{2\pi}{w}j\right), \tag{D.38}$$

one can easily show (i) that $\mathfrak{a}_0$ is independent of scale changes a but a constant shift of the image brightness by o is preserved and (ii) that for coefficients $\mathfrak{a}_k$ and $\mathfrak{b}_k$ for $k > 1$ scalings a of the image intensities are preserved, whereas the coefficients are independent of o. Thus, the signature $s_{\mathtt{afc}}$ cannot compensate for linear intensity transformations. As the signature function $s_{\mathtt{zafc}}$ does not contain the first Fourier coefficient $\mathfrak{a}_0$, it is invariant against a constant shift o of the image intensities, but not against a scaling a of intensity values.

D.3. Further Results



(1) Short-term experiments **(2)** Long-term experiments

Figure D.1.: Footprint of the lab of the Bielefeld Computer Engineering group in which the experiments were performed. The light-gray rectangles depict the experimental areas for the short-term experiments (subfigure (1)) and the long-term experiments (subfigure (2)); the trajectory plots given in figures D.2 and D.3 are restricted to these areas. The black cross and the thick marks the positions of the active tracker; the positions of the neon lamps on the lab's ceiling are visualized by dashed lines (subfigure (1) only). The measuring lines at the right lower corner show a length of 1 m.

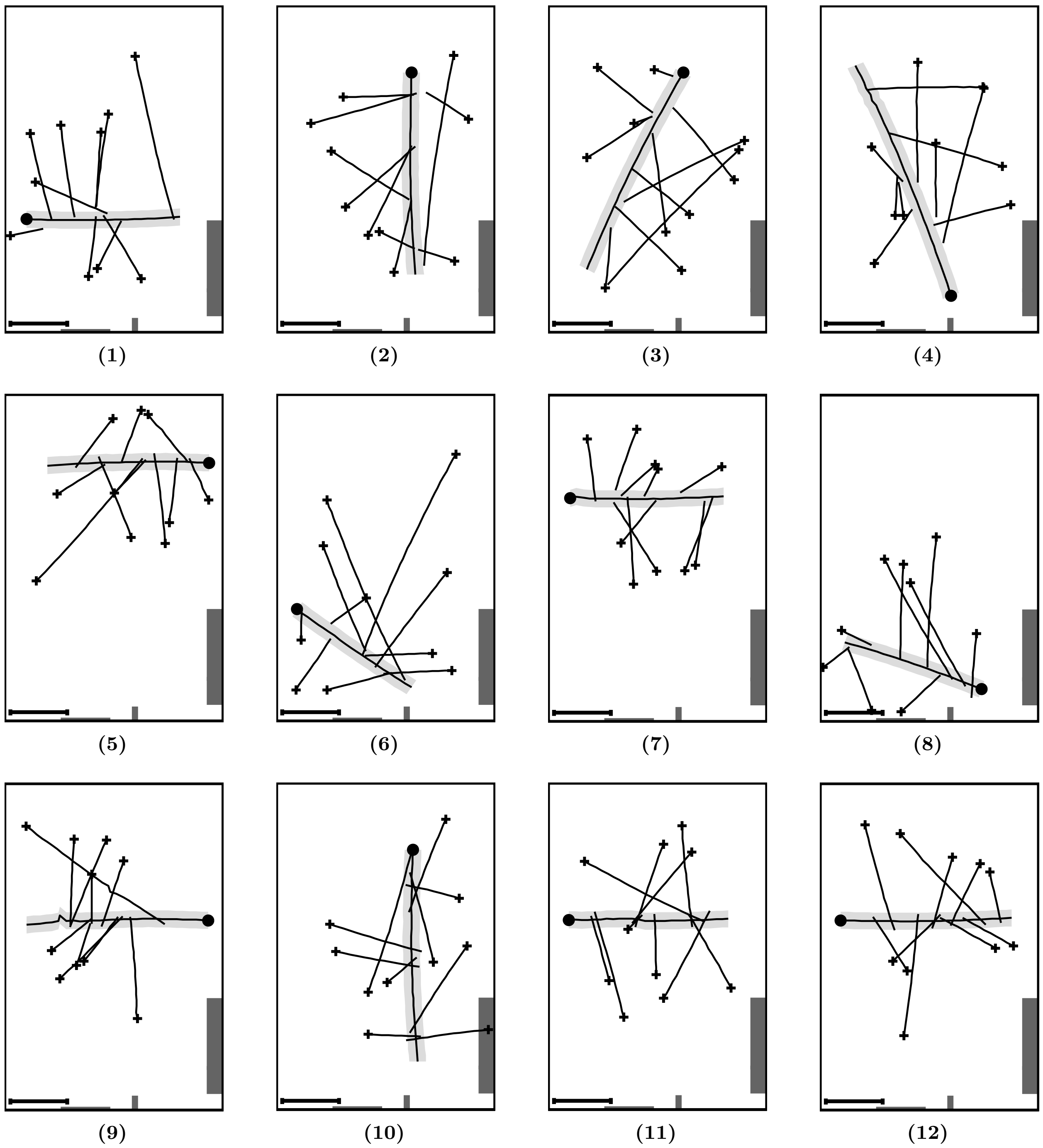


Figure D.2.: Trajectories of short-term experiments. Subfigures (1) to (12) contain the results of the twelve different short-term experiments. Each subfigure shows the rectangle highlighted in figure D.1.1; the measuring line at the left lower corner shows a length of 1 m. In the figures, the area of correct loop-closure detection is depicted by the light-gray area along the reference lane. The start positions of the reference lane and the test trials are marked by filled circles and crosses, respectively. The end positions of the test trials corresponds to the positions of the last stored snapshot and not to the robot's true end positions. If the robot was stopped, the final position plotted in the figures is not the robot's true position but the position of the last snapshot before the robot was stopped. Thus, trajectories recored while the robot approaches the reference lane can be up to 10 cm longer than depicted here. This drawback of the currently used software to conduct real-robot experiments only affects the visualization of the trajectories but not the evaluation of the experiments. It will be improved in future work.

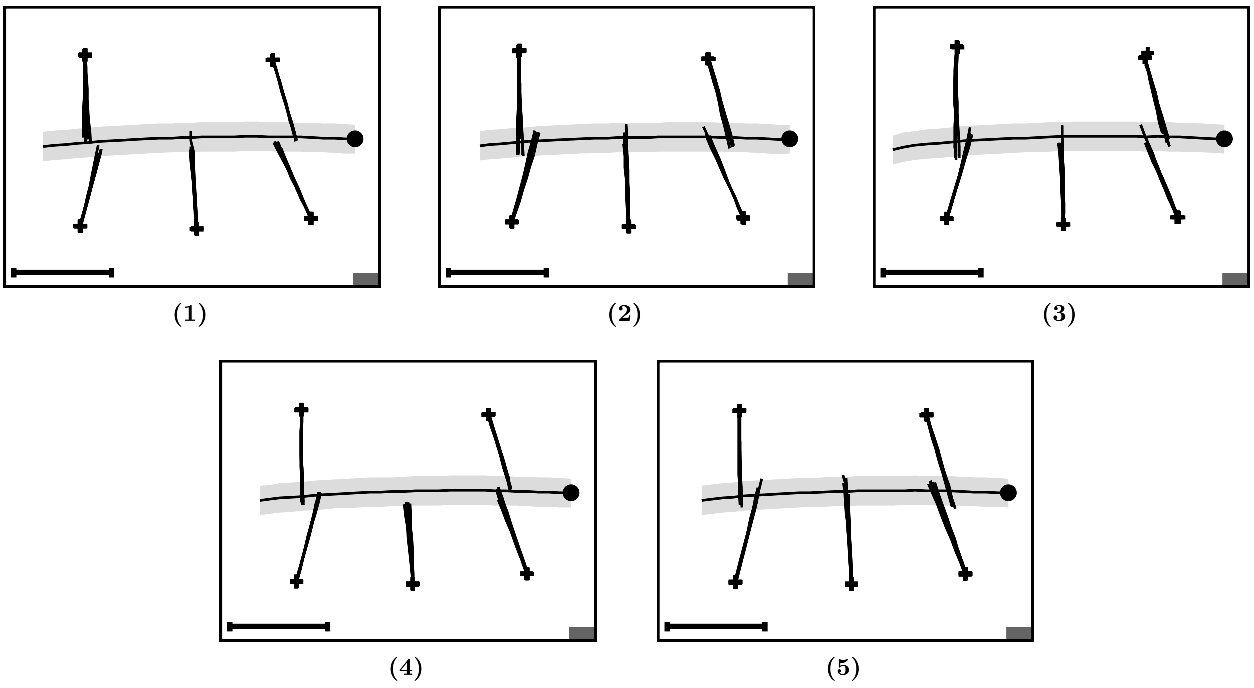


Figure D.3.: Trajectories of long-term experiments. Subfigures (1) to (5) contain the results of the five different long-term experiments. Each subfigure shows the rectangle marked in figure D.1.2; the measuring line at the left lower corner shows a length of 1 m. The area of correct loop-closure detection is visualized by the light-gray bar. Start positions of the reference lane and test trials are marked by a black circles and black crosses, respectively. The end positions of the test trials corresponds to the positions of the last stored snapshot and not to the robot's true end positions. Thus, trajectories recored while the robot approaches the reference lane can be up to 10 cm longer than depicted here. This drawback of the currently used software to conduct real-robot experiments only affects the visualization of the trajectories but not the evaluation of the experiments. It will be improved in future work.

Bibliography

[1] C. D. Ahrens. *Meteorology Today. Introduction to Weather and Climate.* 10th ed. Cengage Learning, 2012 (↗ p. 202).

[2] A. Alahi, R. Ortiz, and P. Vandergheynst. “FREAK: Fast Retina Keypoint.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 12).* 2012, pp. 510–517 (↗ p. 41).

[3] F. Amorós, L. Payá, Ó. Reinoso, and L. M. Jiménez. “Comparison of Global-Appearance Techniques Applied to Visual Map Building and Localization.” In: *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP 12).* 2012, pp. 395–398 (↗ p. 32).

[4] P. Andersen, R. Morris, D. Amaral, T. Bliss, and J. O’Keefe. *The Hippocampus Book.* Oxford University Press, 2007 (↗ pp. 66, 86).

[5] H. Andreasson, T. Duckett, and A. J. Lilienthal. “Mini-SLAM: Minimalistic Visual SLAM in Large-Scale Environments Based on a New Interpretation of Image Similarity.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 07).* 2007, pp. 4096–4101 (↗ pp. 26, 36, 39, 46, 77).

[6] H. Andreasson, T. Duckett, and A. J. Lilienthal. “A Minimalistic Approach to Appearance-Based Visual SLAM.” In: *IEEE Transactions on Robotics* 24.5 (2008), pp. 991–1001 (↗ pp. 26, 36, 39, 46, 77).

[7] H. Andreasson, A. Treptow, and T. Duckett. “Localization for Mobile Robots Using Panoramic Vision, Local Features and Particle Filter.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 05).* 2005, pp. 3348–3353 (↗ pp. 26, 35, 39, 78).

[8] H. Andreasson, A. Treptow, and T. Duckett. “Self-Localization in Non-Stationary Environments Using Omni-Directional Vision.” In: *Robotics and Autonomous Systems* 55.7 (2007), pp. 541–551 (↗ pp. 26, 35, 39, 78, 89).

[9] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. “An Introduction to MCMC for Machine Learning.” In: *Machine Learning* 50.1 (2003), pp. 5–43 (↗ p. 39).

[10] D. Anguelov, C. Dulong, D. Filip, C. Frueh, S. Lafon, R. Lyon, A. Ogale, L. Vincent, and J. Weaver. “Google Street View: Capturing the World at Street Level.” In: *Computer* 43.6 (2010), pp. 32–38 (↗ p. 26).

[11] G. Antonelli, T. I. Fossen, and D. R. Yoerger. “Underwater Robotics.” In: *Springer Handbook of Robotics.* Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 987–1008 (↗ p. 1).

[12] M. Aranda, G. López-Nicolás, and Sagüés. “Omnidirectional Visual Homing Using the 1D Trifocal Tensor.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 10).* May 2010, pp. 2444–2450 (↗ p. 57).

[13] P. Arena, S. De Fiore, L. Fortuna, L. Nicolosi, G. Vagliasindi, and L. Patane. “Visual Homing: Experimental Results on an Autonomous Robot.” In: *Proceedings of the European Conference on Circuit Theory and Design (ECCTD 07).* Aug. 2007, pp. 304–307 (↗ p. 52).

[14] A. A. Argyros, K. E. Bekris, S. C. Orphanoudakis, and L. E. Kavraki. “Robot Homing by Exploiting Panoramic Vision.” In: *Autonomous Robots* 19.1 (July 2005), pp. 7–25 (↗ pp. 26, 35, 57, 58, 82).

[15] Z. Arican and P. Frossard. “Dense Disparity Estimation from Omnidirectional Images.” In: *Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS 07).* 2007, pp. 399–404 (↗ p. 74).

[16] R. Arora and H. Parthasarathy. “Navigation Using a Spherical Camera.” In: *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR 08).* Dec. 2008, pp. 1–4 (↗ p. 53).

[17] C. Arth, M. Klopschitz, G. Reitmayr, and D. Schmalstieg. “Real-Time Self-Localization from Panoramic Images on Mobile Devices.” In: *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality (ISMAR 11).* 2011, pp. 37–46 (↗ p. 85).

[18] A. Ascani, E. Frontoni, A. Mancini, and P. Zingaretti. “Feature Group Matching for Appearance-Based Localization.” In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 08).* 2008, pp. 3933–3938 (↗ pp. 35, 39, 82).

[19] P. Aschwanden and W. Guggenbühl. “Experimental Results from a Comparative Study on Correlation-Type Registration Algorithms.” In: *Robust Computer Vision: Quality of Vision Algorithms.* Ed. by W. Förstner and S. Ruwiedel. Wichmann, 1992, pp. 268–288 (↗ pp. 22, 30, 33, 42, 45, 119, 123).

[20] A. Avarguès-Weber, N. Deisig, and M. Giurfa. “Visual Cognition in Social Insects.” In: *Annual Review of Entomology* 56 (1 2011), pp. 423–443 (↗ p. 42).

[21] B. Baddeley, P. Graham, P. Husbands, and A. Philippides. “A Model of Ant Route Navigation Driven by Scene Familiarity.” In: *PLoS Computational Biology* 8.1 (2012), e1002336 (↗ pp. 83, 84, 86).

[22] B. Baddeley, P. Graham, A. Philippides, and P. Husbands. “Holistic Visual Encoding of Ant-Like Routes: Navigation Without Waypoints.” In: *Adaptive Behavior* 19.1 (2011), pp. 3–15 (↗ pp. 83–86).

[23] B. Baddeley, P. Graham, A. Philippides, and P. Husbands. “Models of Visually Guided Routes in Ants: Embodiment Simplifies Route Acquisition.” In: *Intelligent Robotics and Applications.* Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. P. Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, and G. Weikum. Vol. 7102. Lecture Notes in Computer Science (LNCS). Springer, 2011, pp. 75–84 (↗ pp. 83, 84, 86).

[24] B. Baddeley, P. Graham, A. Philippides, and P. Husbands. “Route Learning Through Classification.” In: *Proceedings of the International Symposium on AI-Inspired Biology (AISB 10).* 2010, pp. 14–19 (↗ pp. 83–85).

[25] T. Bailey and H. Durrant-Whyte. “Simultaneous Localization and Mapping (SLAM). Part II: State of the Art.” In: *IEEE Robotics and Automation Magazine* 13.3 (2006), pp. 108–117 (↗ pp. 5, 17, 39, 40, 65–67).

[26] S. Baker and S. K. Nayar. “A Theory of Single-Viewpoint Catadioptric Image Formation.” In: *International Journal of Computer Vision* 32.5 (1999), pp. 175–196 (↗ pp. 24, 25).

[27] B. Balaguer, S. Balakirsky, S. Carpin, and A. Visser. “Evaluating Maps Produced by Urban Search and Rescue Robots: Lessons Learned from Robocup.” In: *Autonomous Robots* 27.4 (2009), pp. 449–464 (↗ p. 88).

[28] A. Barrera, A. Cáceres, A. Weitzenfeld, and V. Ramirez-Amaya. “Comparative Experimental Studies on Spatial Memory and Learning in Rats and Robots.” In: *Journal of Intelligent and Robotic Systems* 63.3 (2011), pp. 361–397 (↗ p. 86).

[29] A. Barrera and A. Weitzenfeld. “Biologically-Inspired Robot Spatial Cognition Based on Rat Neurophysiological Studies.” In: *Autonomous Robots* 25.1 (2008), pp. 147–169 (↗ p. 86).

[30] J. L. Barron, D. J. Fleet, and S. Beauchemin. “Performance of Optical Flow Techniques.” In: *International Journal of Computer Vision* 12.1 (1994), pp. 43–77 (↗ pp. 28, 54, 219, 223).

[31] K. Basten and H. A. Mallot. “Simulated Visual Homing in Desert Ant Environments: Efficiency of Skyline Cues.” In: *Biological Cybernetics* 102 (2010), pp. 413–425 (↗ pp. 42, 52, 54, 60, 212).

[32] E. Batschelet. *Circular Statistics in Biology.* Academic Press, 1981 (↗ pp. 101, 142, 167).

[33] H. Bay, T. Tuytelaars, and L. Van Gool. “SURF: SPeeded Up Robust Features.” In: *Proceedings of the European Conference on Computer Vision (ECCV 06).* Ed. by A. Leonardis, H. Bischof, and A. Prinz. Vol. 3951. Lecture Notes in Computer Science (LNCS). Springer, 2006, pp. 346–359 (↗ pp. 35, 41).

[34] S. Bazeille and D. Filliat. “Incremental Topo-Metric SLAM Using Vision and Robot Odometry.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 11).* 2011, pp. 4067–4073 (↗ pp. 35, 36, 39, 77).

[35] S. S. Beauchemin and J. L. Barron. “The Computation of Optical Flow.” In: *ACM Computing Surveys* 27.3 (1995), pp. 433–466 (↗ p. 58).

[36] H. M. Becerra, G. López-Nicolás, and C. Sagüés. “Omnidirectional Visual Control of Mobile Robots Based on the 1D Trifocal Tensor.” In: *Robotics and Autonomous Systems* 58.6 (June 2010), pp. 796–808 (↗ pp. 35, 57).

[37] M. Behnisch. “Visuelles Homing mit adaptiven Verfahren zur Bestimmung des optischen Flusses.” Diploma Thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2007. In German. (↗ pp. 50, 56).

[38] K. E. Bekris, A. A. Argyros, and L. E. Kavraki. “Angle-Based Methods for Mobile Robot Navigation: Reaching the Entire Plane.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 04).* May 2004, pp. 2373–2378 (↗ p. 58).

[39] K. E. Bekris, A. A. Argyros, and L. E. Kavraki. “Exploiting Panoramic Vision for Bearing-Only Robot Homing.” In: *Imaging Beyond the Pinhole Camera.* Ed. by K. Daniilidis and R. Klette. Vol. 33. Computational Imaging. Springer, 2006, pp. 229–251 (↗ pp. 57, 58).

[40] R. Benosman and S. B. Kang, eds. *Panoramic Vision: Sensors, Theory, and Applications.* Springer, 2001 (↗ pp. 23–26, 36).

[41] K. Berns and E. von Puttkammer. *Autonomous Land Vehicles: Steps Towards Service Robots.* Vieweg and Teubner, 2009 (↗ pp. 11, 17, 66, 70, 72, 75, 78, 82, 113, 217, 220).

[42] P. Biber, H. Andreasson, T. Duckett, and A. Schilling. “3D Modeling of Indoor Environments by a Mobile Robot with a Laser Scanner and Panoramic Camera.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 04).* Vol. 4. 2004, pp. 3430–3435 (↗ p. 73).

[43] J. Billingsley, A. Visala, and M. Dunn. “Robotics in Agriculture and Forestry.” In: *Springer Handbook of Robotics.* Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 853–869 (↗ pp. 1, 217).

[44] D. M. Binding and F. Labrosse. “Visual Local Navigation Using Warped Panoramic Images.” In: *Proceedings of the Towards Autonomic Robotic Systems (TAROS 06).* Sept. 2006, pp. 19–26 (↗ p. 52).

[45] D. Biro. “Bird Navigation: A Clear View of Magnetoreception.” In: *Current Biology* 20.14 (2010), R595–R596 (↗ pp. 47, 61).

[46] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006 (↗ pp. 16, 33, 35, 39, 56, 83, 212).

[47] B. G. Blundell. *An Introduction to Computer Graphics and Creative 3D-Environments.* Springer, 2008 (↗ p. 130).

[48] A. Bonarini, W. Burgard, G. Fontana, M. Matteucci, D. G. Sorrenti, and J. D. Tardos. “RAWSEEDS. Robotics Advancement through Web-publishing of Sensorial and Elaborated Extensive Data Sets.” In: *In Proceedings of the IROS 06 Workshop on Benchmarks in Robotics Research*. 2006 (↗ p. 88).

[49] P. Bonasso and D. Miller, eds. *Instantiating Real-World Agents: Papers from the 1993 AAAI Fall Symposium*. AAAI Press, 1994 (↗ p. 14).

[50] V. Bonin, V. Mante, and M. Carandini. “The Suppressive Field of Neurons in Lateral Geniculate Nucleus.” In: *Journal of Neuroscience* 25.47 (2005), pp. 10844–10856 (↗ p. 122).

[51] V. Bonin, V. Mante, and M. Carandini. “The Statistical Computation Underlying Contrast Gain Control.” In: *Journal of Neuroscience* 26.23 (2006), pp. 6346–6353 (↗ p. 122).

[52] F. Bonin-Font, A. Ortiz, and G. Oliver. “Visual Navigation for Mobile Robots. A Survey.” In: 53.3 (2008), pp. 263–296 (↗ p. 17).

[53] J. F. Bonora and D. Gallardo. “Visual Homing Navigation with Two Landmarks: The Balanced Proportional Triangulation Method.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 06)*. Oct. 2006, pp. 2289–2295 (↗ p. 58).

[54] O. Booij, B. Terwijn, Z. Zivkovic, and B. Kröse. “Navigation Using an Appearance Based Topological Map.” In: *Proceedings of the IEEE Intnernational Conference on Robotics and Automation (ICRA 07)*. 2007, pp. 3927–3932 (↗ pp. 26, 36, 68, 81).

[55] O. Booij, Z. Zivkovic, and B. Kröse. “Sampling in Image Space for Vision Based SLAM.” In: *Inside Data Association Workshop (Robotics: Science and Systems Conference)*. 2008 (↗ pp. 26, 35, 36, 68, 81).

[56] O. Booij, Z. Zivkovic, and B. Kröse. “Efficient Data Association for View-Based SLAM Using Connected Dominating Sets.” In: *Robotics and Autonomous Systems* 57 (12 2009), pp. 1225–1234 (↗ pp. 26, 35, 36, 68, 81).

[57] O. Booij, Z. Zivkovic, and B. Kröse. “Efficient Probabilistic Planar Robot Motion Estimation Given Pairs of Images.” In: *Proceedings of Robotics: Science and Systems (RSS 10)*. 2010, P26 (↗ pp. 35, 57, 58).

[58] O. Booij, Z. Zivkovic, and B. Kröse. “Sparse Appearance Based Modeling for Robot Localization.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 06)*. 2006, pp. 1510–1515 (↗ p. 81).

[59] G. Borghi and D. Brugali. “Autonomous Map Learning for A Multisensor Mobile Robot Using Diktiometric Representation and Negotiation Mechanism.” In: *Proceedings of the IEEE International Conference on Advanced Robotics(ICAR 95)*. 1995, pp. 521–528 (↗ p. 67).

[60] B. Böttcher. “Bildvorverarbeitung und Bildähnlichkeitsfunktionen für beleuchtungstolerante Navigation.” Bachelor’s Thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2009. In German. (↗ pp. 17, 22, 117, 123).

[61] J. Y. Bouguet. *Pyramidal Implementation of the Lucas-Kanade Feature Tracker*. Technical report. Microprocessor Research Labs, Intel Corporation, 2000 (↗ p. 55).

[62] R. Boutteau, X. Savatier, J. Y. Ertaud, and B. Mazari. “A 3D Omnidirectional Sensor for Mobile Robot Applications.” In: *Mobile Robots Navigation*. Ed. by A. Barrera. InTech, 2010, pp. 1–25 (↗ pp. 39, 71).

[63] D. P. Bovet and M. Cesati. *Understanding the Linux Kernel*. 3rd ed. O’Reilly, 2005 (↗ p. 142).

[64] M. S. Bowlin, I.-A. Bisson, J. Shamoun-Baranes, J. D. Reichard, N. Sapir, P. P. Marra, T. H. Kunz, D. S. Wilcove, A. Hedenström, C. G. Guglielmo, S. Åkesson, M. Ramenofsky, and M. Wikelski. “Grand Challenges in Migration Biology.” In: *Integrative and Comparative Biology* 50.3 (2010), pp. 261–279 (↗ p. 61).

[65] R. N. Bracewell. *Fourier Analysis and Imaging*. Springer, 2003 (↗ pp. 31, 45, 127, 128, 245).

[66] G. R. Bradski. “Computer Vision Face Tracking for Use in a Perceptual User Interface.” In: *Intel Technology Journal* Q2 (1998) (↗ p. 203).

[67] G. R. Bradski and A. Kaehler. *Learning OpenCv. Computer Vision with the OpenCv Library*. O’Reilly, 2008 (↗ pp. 123, 178).

[68] A. Briggs, Y. Li, D. Scharstein, and M. Wilder. “Robot Navigation Using 1D Panoramic Images.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 06)*. May 2006, pp. 2679–2685 (↗ p. 78).

[69] A. J. Briggs, C. Detweiler, Y. Li, P. C. Mullen, and D. Scharstein. “Matching Scale-Space Features in 1D Panoramas.” In: *Computer Vision and Image Understanding* 103.3 (Sept. 2006), pp. 184–195 (↗ pp. 57, 58).

[70] R. Bunschoten and B. Kröse. “Robust Scene Reconstruction from an Omnidirectional Vision System.” In: *IEEE Transactions on Robotics and Automation* 19.2 (2003), pp. 351–357 (↗ p. 74).

[71] R. Bunschoten and B. Kröse. “Visual Odometry from an Omnidirectional Vision System.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 03)*. 2003, pp. 577–583 (↗ pp. 35, 71, 72).

[72] R. Bunschoten. “Mapping and Localization from a Panoramic Vision Sensor.” PhD thesis. Faculty of Science, University of Amsterdam, Netherlands, 2003 (↗ pp. 23, 24).

[73] R. Bunschoten and B. Kröse. “3D Scene Reconstruction from Cylindrical Panoramic Images.” In: *Robotics and Autonomous Systems* 41.2–3 (2002), pp. 111–118 (↗ pp. 35, 74).

[74] W. Burgard and M. Hebert. “World Modeling.” In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 853–869 (↗ pp. 5, 17, 20, 23, 62, 64, 65, 72, 75).

[75] A. Burke and A. Vardy. “Visual Compass Methods for Robot Navigation.” In: *Proceedings of the Newfoundland Conference on Electrical and Computer Engineering*. 2006 (↗ pp. 45, 47, 129).

[76] P. Buschka and A. Saffiotti. “Some Notes on the Use of Hybrid Maps for Mobile Robots.” In: *Proceedings of the International Conference on Intelligent Autonomous Systems (IAS 04)*. 2004, pp. 547–556 (↗ p. 67).

[77] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. “BRIEF: Binary Robust Independent Elementary Features.” In: *Proceedings of the European Conference on Computer Vision (ECCV 10)*. Ed. by K. Daniilidis, P. Maragos, and N. Paragios. Vol. 6312. Lecture Notes in Computer Science (LNCS). Springer, 2010, pp. 778–792 (↗ p. 41).

[78] K. Caluwaerts, M. Staffa, S. N’Guyen, C. Grand, L. Dollé, A. Favre-Félix, B. Girard, and M. Khamassi. “A Biologically Inspired Meta-Control Navigation System for the Psikharpax Rat Robot.” In: *Bioinspiration and Biomimetics* 7.2 (2012), p. 025009 (↗ p. 86).

[79] J. Campbell, R. Sukthankar, and I. Nourbakhsh. “Techniques for Evaluating Optical Flow for Visual Odometry in Extreme Terrain.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 04)*. 2004, pp. 3704–3711 (↗ p. 107).

[80] J. Campbell, R. Sukthankar, I. Nourbakhsh, and A. Pahwa. “A Robust Visual Odometry and Precipice Detection System Using Consumer-grade Monocular Vision.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 05)*. 2005, pp. 3421–3427 (↗ p. 107).

[81] M. Carandini, J. B. Demb, V. Mante, D. J. Tolhurst, Y. Dan, B. A. Olshausen, J. L. Gallant, and N. C. Rust. “Do We Know What the Early Visual System Does?” In: *The Journal of Neuroscience* 25 (2005), pp. 10577–10597 (↗ p. 122).

[82] G. Caron, E. Marchand, and E.-M. Mouaddib. “Tracking Planes in Omnidirectional Stereovision.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 11)*. 2011, pp. 6306–6311 (↗ pp. 35, 71, 72).

[83] G. Caron and E.-M. Mouaddib. “Vertical Line Matching for Omnidirectional Stereovision Images.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 09)*. 2009, pp. 2787–2792 (↗ p. 73).

[84] G. Caron, E.-M. Mouaddib, and E. Marchand. “3D Model Based Tracking for Omnidirectional Vision: A New Spherical Approach.” In: *Robotics and Autonomous Systems* 60.8 (2012), pp. 1056–1068 (↗ pp. 26, 35, 71, 72).

[85] G. Carrera, A. Angeli, and A. J. Davison. “SLAM-Based Automatic Extrinsic Calibration of a Multi-Camera Rig.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 11)*. 2011, pp. 2652–2659 (↗ p. 26).

[86] B. A. Cartwright and T. S. Collett. “Landmark Maps for Honey Bees.” In: *Biological Cybernetics* 57 (1–2 1987), pp. 85–93 (↗ p. 60).

[87] B. Cartwright and T. Collett. “Landmark Learning in Bees.” In: *Journal of Comparative Physiology A* 151.4 (1983), pp. 521–543 (↗ pp. 42, 47, 60).

[88] F. Cepolina, R. C. Michelini, R. P. Razzoli, and M. Zoppi. “Gecko, a Climbing Robot for Walls Cleaning.” In: *Proceedings of the International Workshop on Advances in Service Robotics (ASER 03)*. 2003 (↗ p. 10).

[89] S. Ceriani, G. Fontana, A. Giusti, D. Marzorati, M. Matteucci, D. Migliore, D. Rizzi, D. G. Sorrenti, and P. Taddei. “Rawseeds Ground Truth Collection Systems for Indoor Self-Localization and Mapping.” In: *Autonomous Robots* 27.4 (2009), pp. 353–371 (↗ p. 88).

[90] J. S. Chahl and M. V. Srinivasan. “Reflective Surfaces for Panoramic Imaging.” In: *Applied Optics* 36.31 (1997), pp. 8275–8285 (↗ p. 26).

[91] S. Chambon and A. Crouzil. “Dense Matching Using Correlation: New Measures That Are Robust Near Occlusions.” In: *Proceedings of the British Machine Vision Conference (BMVC 03)*. 2003, pp. 15.1–15.10 (↗ pp. 22, 30, 33, 42, 45, 119, 123).

[92] S. Chambon and A. Crouzil. “Similarity Measures for Image Matching Despite Occlusions in Stereo Matching.” In: *Pattern Recognition* 44.9 (2011), pp. 2063–2075 (↗ pp. 22, 30, 33, 42, 45, 119, 123, 218).

[93] S. Chameron, B. Schatz, I. Pastergue-Ruiz, G. Beugnon, and T. S. Collett. “The Learning of a Sequence of Visual Patterns by the Ant Cataglyphis Cursor.” In: *Proceedings of the Royal Society B: Biological Sciences* 265.1412 (1998), pp. 2309–2313 (↗ p. 86).

[94] C. Charron, O. Labbani-Igbida, and E. Mouaddib. “On Building Omnidirectional Image Signatures Using Haar Invariant Features. Application to the Localization of Robots.” In: *Advanced Concepts for Intelligent Vision Systems*. Ed. by J. Blanc-Talon, W. Philips, D. Popescu, and P. Scheunders. Vol. 4179. Lecture Notes in Computer Science (LNCS). Springer, 2006, pp. 1099–1110 (↗ pp. 32, 82, 170).

[95] F. Chaumette and S. Hutchinson. “Visual Servo Control. Part I: Basic Approaches.” In: *IEEE Robotics and Automation Magazine* 13.4 (2006), pp. 82–90 (↗ p. 49).

[96] F. Chaumette and S. Hutchinson. “Visual Servo Control. Part II: Advanced Approaches.” In: *IEEE Robotics and Automation Magazine* 14.1 (2007), pp. 109–118 (↗ p. 49).

[97] F. Chaumette and S. Hutchinson. “Visual Servoing and Visual Tracking.” In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 563–583 (↗ p. 49).

[98] K.-Y. Chen, C.-C. Chien, W.-L. Chang, and J.-T. Teng. “An Integrated Color and Hand Gesture Recognition Approach for an Autonomous Mobile Robot.” In: *Proceedings of the IEEE International Congress on Image and Signal Processing (CISP 10)*. Vol. 5. 2010, pp. 2496–2500 (↗ pp. 29, 224).

[99] Z. Chen, J. Samarabandu, and R. Rodrigo. “Recent Advances in Simultaneous Localization and Mapping.” In: *Advanced Robotics* 21.3–4 (2007), pp. 233–265 (↗ pp. 17, 20, 22, 40, 62, 65, 66).

[100] K. Cheng. “A Purely Geometric Module in the Rat’s Spatial Representation.” In: *Cognition* 23 (1986), pp. 149–178 (↗ pp. 42, 61).

[101] K. Cheng. “Whither Geometry? Troubles of the Geometric Module.” In: *Trends in Cognitive Science* 12.9 (2008), pp. 355–361 (↗ pp. 42, 61).

[102] K. Cheng, A. Narendra, S. Sommer, and R. Wehner. “Traveling in Clutter: Navigation in the Central Australian Desert Ant Melophorus Bagoti.” In: *Behavioural Processes* 80.3 (2009), pp. 261–268 (↗ p. 86).

[103] A. Cheung, W. Stürzl, J. Zeil, and K. Cheng. “The Information Content of Panoramic Images II. View-Based Navigation in Nonrectangular Experimental Arenas.” In: *Journal of Experimental Psychology: Animal Behavior Processes* 34.1 (2008), pp. 15–30 (↗ pp. 43, 52, 61).

[104] A.-S. Chiang, C.-Y. Lin, C.-C. Chuang, H.-M. Chang, C.-H. Hsieh, C.-W. Yeh, C.-T. Shih, J.-J. Wu, G.-T. Wang, Y.-C. Chen, C.-C. Wu, G.-Y. Chen, Y.-T. Ching, P.-C. Lee, C.-Y. Lin, H.-H. Lin, C.-C. Wu, H.-W. Hsu, Y.-A. Huang, J.-Y. Chen, H.-J. Chiang, C.-F. Lu, R.-F. Ni, C.-Y. Yeh, and J.-K. Hwang. “Three-Dimensional Reconstruction of Brain-Wide Wiring Networks in Drosophila at Single-Cell Resolution.” In: *Current Biology* 21.1 (2011), pp. 1–11 (↗ p. 86).

[105] L. Chittka and J. Niven. “Are Bigger Brains Better?” In: *Current Biology* 19.21 (2009), R995–R1008 (↗ p. 86).

[106] K. Choi, J. Park, Y. Kim, and H. Lee. “Monocular SLAM with Undelayed Initialization for an Indoor Robot.” In: *Robotics and Autonomous Systems* 60.6 (2012), pp. 841–851 (↗ p. 15).

[107] Y. H. Choi, T. K. Lee, S. Baek, and S. Y. Oh. “Online Complete Coverage Path Planning for Mobile Robots Based on Linked Spiral Paths Using Constrained Inverse Distance Transform.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2009, pp. 5788–5793 (↗ p. 15).

[108] E. K. P. Chong and S. H. Żak. *An Introduction to Optimization*. 3rd ed. Wiley, 2008 (↗ p. 178).

[109] H. Choset. “Coverage for Robotics. A Survey of Recent Results.” In: *Annals of Mathematics and Artificial Intelligence* 31.1-4 (2001), pp. 113–126 (↗ p. 15).

[110] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun. *Principles of Robot Motion*. MIT Press, 2005 (↗ pp. 5, 15, 17, 39, 40, 64–66, 75).

[111] W. Chung, G. Kim, M. Kim, and C. Lee. “Integrated Navigation System for Indoor Service Robots in Large-Scale Environments.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 04)*. 2004, pp. 5099–5104 (↗ p. 10).

[112] W. Chung, G. Kim, and M. Kim. “Development of the Multi-Functional Indoor Service Robot PSR Systems.” In: *Autonomous Robots* 22.1 (2007), pp. 1–17 (↗ p. 10).

[113] D. Churchill and A. Vardy. “Homing in Scale Space.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 08)*. Sept. 2008, pp. 1307–1312 (↗ pp. 35, 58).

[114] D. Churchill and A. Vardy. “An Orientation Invariant Visual Homing Algorithm.” In: *Journal of Intelligent and Robotic Systems* 71.1 (2013), pp. 3–29 (↗ pp. 35, 58).

[115] M. Collett and T. S. Collett. “Insect Navigation: No Map at the End of the Trail?” In: *Current Biology* 16.2 (2006), R48–R51 (↗ p. 86).

[116] D. Comaniciu and P. Meer. “Mean Shift: A Robust Approach Toward Feature Space Analysis.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.5 (2002), pp. 603–619 (↗ pp. 106, 107).

[117] D. Comaniciu, V. Ramesh, and P. Meer. “Real-Time Tracking of Non-Rigid Objects Using Mean Shift.” In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (ICPR 00)*. Vol. 2. 2000, p. 142 (↗ p. 176).

[118] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson. *Introduction to Algorithms*. 2nd ed. McGraw-Hill, 2001 (↗ pp. 78, 82, 221).

[119] F. R. Correa and J. Okamoto. “Omnidirectional Stereovision System for Occupancy Grid.” In: *Proceedings of the IEEE International Conference on Advanced Robotics (ICAR 05)*. 2005, pp. 628–634 (↗ pp. 40, 74, 75).

[120] M. Corsini, M. Dellepiane, F. Ponchio, and R. Scopigno. “Image-to-Geometry Registration: A Mutual Information Method Exploiting Illumination-Related Geometric Properties.” In: 28.7 (2009), pp. 1755–1764 (↗ p. 128).

[121] J. Courbon, Y. Mezouar, and P. Martinet. “Indoor Navigation of a Non-Holonomic Mobile Robot Using a Visual Memory.” In: *Autonomous Robots* 25.3 (Oct. 2008), pp. 253–266 (↗ p. 83).

[122] T. M. Cover and J. A. Cover. *Elements of Information Theory.* 2nd ed. Wiley-Interscience, 2006 (↗ pp. 127, 138).

[123] I. J. Cox, S. Roy, and S. L. Hingorani. “Dynamic Histogram Warping of Image Pairs for Constant Image Brightness.” In: *Proceedings of the IEEE International Conference on Image Processing (ICCV 95).* 1995, pp. 366–369 (↗ pp. 167, 218).

[124] H. Cruse. “A Recurrent Network for Landmark-Based Navigation.” In: *Biological Cybernetics* 88.6 (2003), pp. 425–437 (↗ p. 50).

[125] H. Cruse and R. Wehner. “No Need for a Cognitive Map: Decentralized Memory for Insect Navigation.” In: *PLoS Computational Biology* 7.3 (2011), e1002009 (↗ p. 86).

[126] M. Cummins and P. Newman. “FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance.” In: *International Journal of Robotics Research* 27.6 (2008), pp. 647–665 (↗ pp. 81, 88).

[127] M. Cummins and P. Newman. “Appearance-Only SLAM at Large Scale with FAB-MAP 2.0.” In: *International Journal of Robotics Research* 9.30 (2010), pp. 1100–1123 (↗ pp. 35, 81, 85, 87).

[128] M. Cummins and P. Newman. “Highly Scalable Appearance-Only SLAM — FAB-MAP 2.0.” In: *Proceedings of Robotics: Science and Systems (RSS 10).* 2010, P39 (↗ pp. 35, 81, 85).

[129] H.-J. Dahmen, M. O. Franz, and H. G. Krapp. “Extracting Egomotion from Optic Flow: Limits of Accuracy and Neural Matched Filters.” In: *Motion Vision. Computational, Neural, and Ecological Constraints.* Ed. by J. M. Zanker and J. Zeil. Springer, 2001, pp. 144–168 (↗ p. 45).

[130] K. Daniilidis and J. O. Eklundh. “3-D Vision and Recognition.” In: *Springer Handbook of Robotics.* Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 543–562 (↗ pp. 3, 17, 35, 48, 65, 66, 70).

[131] K. Daniilidis and R. Klette, eds. *Imaging Beyond the Pinhole Camera.* Vol. 33. Computational Imaging. Springer, 2006 (↗ pp. 23, 24, 26, 36).

[132] R. Datta, D. Joshi, J. Li, and J. Z. Wang. “Image Retrieval: Ideas, Influences, and Trends of the New Age.” In: *ACM Computing Surveys* 40.2 (2008), pp. 1–60 (↗ pp. 31, 172, 178, 209).

[133] F. Dayoub, G. Cielniak, and T. Duckett. “A Sparse Hybrid Map for Vision-Guided Mobile Robots.” In: *Proceedings of the European Conference on Mobile Robots (ECMR 11).* 2011, pp. 213–218 (↗ pp. 39, 57, 77, 89).

[134] F. Dayoub, G. Cielniak, and T. Duckett. “Long-Term Experiments with an Adaptive Spherical View Representation for Navigation in Changing Environments.” In: *Robotics and Autonomous Systems* 59.5 (2011), pp. 285–295 (↗ pp. 39, 57, 68, 77, 89).

[135] F. Dayoub, T. Duckett, and G. Cielniak. “An Adaptive Spherical View Representation for Navigation in Changing Environments.” In: *Proceedings of the European Conference on Mobile Robots (ECMR 09).* 2009, pp. 1–6 (↗ pp. 67, 77, 89).

[136] D. Dederscheck, H. Friedrich, C. Lenhart, M. Zahn, and R. Mester. “Featuring Optical Rails. View-Based Robot Guidance Using Orientation Features on the Sphere.” In: *Proceedings of the International Conference on Computer Vision (ICCV 09).* 2009, pp. 2156–2163 (↗ pp. 32, 53, 82, 83, 89).

[137] D. Dederscheck, H. Friedrich, C. Lenhart, J. Penc, E. Rosert, M. Scherer, and R. Mester. “Running on Optical Rails: Theory, Implementation and Testing of Omnidirectional View-Based Point-to-Point Navigation.” In: *Proceedings of the Workshop on Omnidirectional Vision, Camera Networks, and Non-Classical Cameras.* 2008 (↗ pp. 32, 53, 82, 83).

[138] D. Dederscheck, M. Zahn, H. Friedrich, and R. Mester. “Slicing the View. Occlusion-Aware View-Based Robot Navigation.” In: *Pattern Recognition.* Ed. by M. Goesele, S. Roth, A. Kuijper, B. Schiele, and K. Schindler. Vol. 6376. Lecture Notes in Computer Science (LNCS). Springer, 2010, pp. 111–120 (↗ pp. 32, 53, 82, 83, 89).

[139] G. N. DeSouza and A. C. Kak. “Vision for Mobile Robot Navigation: A Survey.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.2 (2002), pp. 237–267 (↗ pp. 17, 70).

[140] J. Diard, P. Bessière, and E. Mazer. “A Survey of Probabilistic Models Using the Bayesian Programming Methodology as a Unifying Framework.” In: *Proceedings of the IEEE International Conference on Computational Intelligence, Robotics and Autonomous Systems (CIRAS 03).* 2003 (↗ pp. 39, 40).

[141] L. Dittmar. “Static and Dynamic Snapshots for Goal Localization in Insects?” In: *Communicative and Integrative Biology* 4.1 (Jan. 2011), pp. 17–20 (↗ pp. 42, 60).

[142] L. Dittmar, W. Stürzl, E. Baird, N. Boeddeker, and M. Egelhaaf. “Goal Seeking in Honeybees: Matching of Optic Flow Snapshots?” In: *Journal of Experimental Biology* 213.17 (2010) (↗ pp. 42, 60).

[143] R. C. Dorf and R. H. Bishop. *Modern Control Systems.* 12th ed. Pearson, 2011 (↗ pp. 49, 58).

[144] P. Doubek and T. Svoboda. “Reliable 3D Reconstruction from a Few Catadioptric Images.” In: *Proceedings of the Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras (OMNIVIS 02).* 2002, pp. 71–78 (↗ p. 73).

[145] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. “Rao-Blackwellized Particle Filtering for Dynamic Bayesian Networks.” In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI 00).* 2000, pp. 176–183 (↗ p. 39).

[146] J. R. Driscoll and D. M. Healy. “Computing Fourier Transform and Convolutions on the 2-Sphere.” In: *Advances in Applied Mathematics* 15.2 (1994), pp. 202–250 (↗ pp. 45, 53).

[147] C. Drocourt, L. Delahoche, B. Marhic, and A. Clerentin. “Simultaneous Localization and Map Construction Method Using Omnidirectional Stereoscopic Information.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 02)*. 2002, pp. 894–899 (↗ p. 71).

[148] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. 2nd ed. Wiley-Interscience, 2001 (↗ pp. 35, 56, 83, 138, 139, 182, 212).

[149] G. Dudek and M. Jenkin. “Inertial Sensors, GPS, and Odometry.” In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 477–490 (↗ p. 217).

[150] H. Durrant-Whyte and T. Bailey. “Simultaneous Localization and Mapping (SLAM). Part I: The Essential Algorithms.” In: *IEEE Robotics and Automation Magazine* 13.2 (2006), pp. 99–110 (↗ pp. 5, 17, 39, 40, 65, 66).

[151] H. Durrant-Whyte and T. C. Henderson. “Multisensor Data Fusion.” In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 871–889 (↗ pp. 39, 40).

[152] Y. Edan, S. Han, and N. Kondo. “Automation and Agriculture.” In: *Handbook of Automation*. Ed. by S. Y. Nof. Springer, 2009, pp. 1253–1264 (↗ pp. 1, 217).

[153] S. Edelman. “Computational Theories of Object Recognition.” In: *Trends in Cognitive Sciences* 1.8 (1997), pp. 296–304 (↗ p. 20).

[154] N. Elkmann, H. Althoff, S. Kutzner, T. Stürze, J. Saenz, and B. Reimann. “Development of Fully Automatic Inspection Systems for Large Underground Concrete Pipes Partially Filled with Wastewater.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 07)*. 2007, pp. 130–135 (↗ p. 10).

[155] N. Elkmann, D. Kunst, T. Krüger, M. Lucke, T. Böhme, T. Felsch, and T. Stürze. “Siriusc — Facade Cleaning Robot for a High-Rise Building in Munich, Germany.” In: *Climbing and Walking Robots. Proceedings of the International Conference on Climbing and Walking Robots (CLAWAR 04)*. Ed. by M. Armada and P. G. de Santo. Springer, 2005, pp. 1033–1040 (↗ p. 10).

[156] N. Elkmann, B. Reimann, E. Schulenburg, and H. Althoff. “Automated Inspection System for Large Underground Concrete Pipes Under Operating Conditions.” In: *Field and Service Robotics*. Vol. 25. Springer Tracts in Advanced Robotics (STAR). Springer, 2006, pp. 567–578 (↗ p. 10).

[157] N. Elkmann, J. Hortig, and M. Fritzsche. “Cleaning Automation.” In: *Handbook of Automation*. Ed. by S. Y. Nof. Springer, 2009, pp. 1253–1264 (↗ pp. 1, 9–11).

[158] M. Ende. *Momo*. Translated by J. M. Brownjohn. Puffin Books, 2009. Originally published in German by K. Thienemanns Verlag, 1973. (↗ p. vii).

[159] J. Engelbrecht, M. Höner, A. Lemme, and I. Moutogiorgos. *Visuelles Trackingsystem*. Project thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2009. In German. (↗ p. 91).

[160] S. P. Engelson and D. V. McDermott. “Error Correction in Mobile Robot Map Learning.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 92)*. 1992, pp. 2555–2560 (↗ p. 67).

[161] S. P. Engelson and D. V. McDermott. “Maps Considered as Adaptive Planning Resources.” In: *AAAI Fall Symposium on Applications of Artificial Intelligence to Real-World Autonomous Mobile Robots*. 1992, pp. 36–44 (↗ p. 67).

[162] C. Enroth-Cugell and J. G. Robson. “The Contrast Sensitivity of Retinal Ganglion Cells of the Cat.” In: *Journal of Physiology* 187.3 (1966), pp. 517–52 (↗ pp. 122, 123).

[163] I. Esteban, O. Booij, J. Dijk, and F. Groen. “On the Bending Problem for Large Scale Mapping.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 09)*. 2009, pp. 1144–1149 (↗ pp. 35, 36, 39, 57, 77).

[164] I. Esteban, O. Booij, Z. Zivkovic, and B. Kröse. “Mapping Large Environments with an Omnivideo Camera.” In: *Proceedings of the International Conference on Simulation, Modelling and Programming of Autonomous Robots (SIMPAR 08)*. 2008, pp. 297–306 (↗ pp. 35, 36, 39, 57, 77).

[165] C. Estrada, J. Neira, and J. D. Tadós. “Hierarchical SLAM: Real-Time Accurate Mapping of Large Environments.” In: *IEEE Transactions on Robotics* 21.4 (2005), pp. 588–596 (↗ p. 67).

[166] R. M. Eustice, H. Singh, and J. J. Leonard. “Exactly Sparse Delayed-State Filters.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 05)*. 2005, pp. 2417–2424 (↗ p. 39).

[167] R. M. Eustice, H. Singh, and J. J. Leonard. “Exactly Sparse Delayed-State Filters for View-Based SLAM.” In: *IEEE Transactions on Robotics* 22.6 (2006), pp. 1100–1114 (↗ p. 39).

[168] T. Fawcett. “An Introduction to ROC Analysis.” In: *Pattern Recognition Letters* 27.8 (2006), pp. 861–874 (↗ pp. 138, 139, 182).

[169] S. N. Ferdaus, A. Vardy, G. Mann, and R. Gosine. “Comparing Global Measures of Image Similarity for Use in Topological Localization of Mobile Robots.” In: *Proceedings of the Canadian Conference on Electrical and Computer Engineering (CCECE 08)*. 2008, pp. 913–918 (↗ pp. 32, 82, 170).

[170] I. Fernández, M. Mazo, J. L. Lázaro, D. Pizarro, E. Santiso, P. Martín, and C. Losada. “Guidance of a Mobile Robot Using an Array of Static Cameras Located in the Environment.” In: *Autonomous Robots* 23.4 (2007), pp. 305–324 (↗ p. 107).

[171] L. Fernández, A. Gil, L. Payá, and Ó. Reinoso. “An Evaluation of Weighting Methods for Appearance-Based Monte-Carlo Localization Using Omnidirectional Images.” In: *Proceedings of the Workshop on Omnidirectional Robot Vision*. 2010, pp. 13–18 (↗ pp. 3, 32, 39, 77, 78, 170).

[172] L. Fernández, L. Payá, Ó. Reinoso, and F. Amorós. “Appearance-Based Visual Odometry with Omnidirectional Images. A Practical Application to Topological Mapping.” In: *Proceedings of the International Conference on Informatics in Control, Automation and Robotics (ICINCO 11)*. 2011, pp. 205–210 (↗ pp. 32, 46, 77, 170).

[173] M. Fiala and A. Basu. “Robot Navigation Using Panoramic Landmark Tracking.” In: *Pattern Recognition* 37.11 (2004), pp. 2195–2215 (↗ p. 57).

[174] P. Filipek and T. Kamiński. “Remote Controlled Mobile Inspection Robot.” In: *Journal of KONES Powertrain and Transport* 18.2 (2011), pp. 129–135 (↗ p. 10).

[175] D. Filliat and J.-A. Meyer. “Map-Based Navigation in Mobile Robots. Part I: A Review of Localization Strategies.” In: *Cognitive Systems Research* 4.4 (2003), pp. 243–282 (↗ pp. 5, 17, 20, 21, 62, 65, 70, 80, 85).

[176] P. Fiorini and E. Prassler. “Cleaning and Household Robots: A Technology Survey.” In: *Autonomous Robots* 9.3 (2000), pp. 227–235 (↗ pp. 10, 11, 14).

[177] S. Fischer. “Visuelle Navigation mit Parametermodellen.” Diploma thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2006. In German. (↗ pp. 53, 172, 173, 210).

[178] M. A. Fischler and R. C. Bolles. “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography.” In: *Communications of the ACM* 24.6 (1981), pp. 381–395 (↗ p. 36).

[179] R. B. Fisher and K. Konolige. “Range Sensors.” In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 521–542 (↗ p. 223).

[180] S. Fleck, F. Busch, P. Biber, W. Strasser, and H. Andreasson. “Omnidirectional 3D Modeling on a Mobile Robot Using Graph Cuts.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 05)*. 2005, pp. 1748–1754 (↗ p. 74).

[181] S. Fleck, F. Busch, P. Biber, and W. Strasser. “Graph Cut Based Panoramic 3D Modeling and Ground Truth Comparison with a Mobile Platform - the WÄgele.” In: *Image and Vision Computing* 27.1-2 (2009), pp. 141–152 (↗ p. 74).

[182] L. Florack. *Image Structure*. Kluwer Academic Publishers, 1997 (↗ p. 59).

[183] J. Fogarty, R. S. Baker, and S. E. Hudson. “Case Studies in the Use of ROC Curve Analysis for Sensor-Based Estimates in Human Computer Interaction.” In: *Proceedings of the Graphics Interface (GI 05)*. 2005, pp. 129–136 (↗ p. 138).

[184] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice in C*. 2nd ed. Addison-Wesley, 1996 (↗ p. 130).

[185] J. Forlizzi and C. DiSalvo. “Service Robots in the Domestic Environment: A Study of the Roomba Vacuum in the Home.” In: *Proceedings of the ACM SIGCHI/SIGART Conference on Human-Robot Interaction (HRI 06)*. 2006, pp. 258–265 (↗ p. 14).

[186] D. A. Forsyth and J. Ponce. “Image-Based Modeling and Rendering.” In: *Computer Vision: A Modern Approach*. 2nd ed. Pearson, 2012, pp. 559–587 (↗ pp. 24, 36, 66, 224).

[187] D. Fox, J. Hightower, L. Liao, D. Schulz, and G. Borriello. “Bayesian Filtering for Location Estimation.” In: *IEEE Pervasive Computing* 2.3 (2003), pp. 24–33 (↗ pp. 39, 40).

[188] D. Fox, J. Hightower, H. Kauz, L. Liao, and D. Patterson. “Bayesian Techniques for Location Estimation.” In: *Proceedings of Workshop on Location-Aware Computing*. 2003, pp. 16–18 (↗ p. 40).

[189] M. O. Franz and H. A. Mallot. “Biomimetic Robot Navigation.” In: *Robotics and Autonomous Systems* 30.1–2 (2000), pp. 133–153 (↗ pp. 3, 17, 19, 20, 47, 48, 50, 60, 80, 81, 83, 85, 213, 216).

[190] M. O. Franz, B. Schölkopf, H. A. Mallot, and H. H. Bülthoff. “Where Did I Take That Snapshot? Scene-based Homing by Image Matching.” In: *Biological Cybernetics* 79.3 (1998), pp. 191–202 (↗ pp. 51, 103, 114).

[191] F. Fraundorfer and D. Scaramuzza. “Visual Odometry. Part II: Matching, Robustness, Optimization, and Applications.” In: *IEEE Robotics and Automation Magazine* 19.2 (2012), pp. 78–90 (↗ pp. 3, 17, 36, 65, 93, 113).

[192] R. A. Frazor and W. S. Geisler. “Local Luminance and Contrast in Natural Images.” In: *Vision Resarch* 46.10 (2006), pp. 1585–1598 (↗ p. 122).

[193] L. Freda and G. Oriolo. “Frontier-Based Probabilistic Strategies for Sensor-Based Exploration.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 05)*. 2005, pp. 3881–3887 (↗ p. 15).

[194] U. Frese. “A Discussion of Simultaneous Localization and Mapping.” In: *Autonomous Robots* 20.1 (2006), pp. 25–42 (↗ pp. 17, 20, 40, 65, 66).

[195] U. Frese, P. Larsson, and T. Duckett. “A Multilevel Relaxation Algorithm for Simultaneous Localization and Mapping.” In: *IEEE Transactions on Robotics* 21.2 (2005), pp. 196–207 (↗ p. 39).

[196] Y. Freund and R. E. Schapire. “A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting.” In: *Proceedings of the European Conference on Computational Learning Theory (EuroCOLT 95)*. Ed. by P. M. B. Vitányi. Vol. 904. Lecture Notes in Computer Science (LNCS). 1995, pp. 23–37 (↗ p. 212).

[197] B. Fritzke. “A Growing Neural Gas Network Learns Topologies.” In: *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*. 1995, pp. 625–632 (↗ p. 212).

[198] E. Frontoni, A. Ascani, A. Mancini, and P. Zingaretti. “Omnidirectional Vision for Robot Localization in Urban Environments.” In: *Proceedings of the International Conference on Simulation, Modeling and Programming of Autonomous Robots (SIMPAR 08)*. 2008, pp. 343–353 (↗ pp. 35, 78).

[199] E. Frontoni, A. Ascani, A. Mancini, and P. Zingaretti. “Performance Metric for Vision Based Robot Localization.” In: *Proceedings of Robotics: Science and Systems (RSS 08)*. 2008 (↗ pp. 35, 82).

[200] E. Frontoni and P. Zingaretti. “An Efficient Similarity Metric for Omnidirectional Vision Sensors.” In: *Robotics and Autonomous Systems* 54.9 (2006), pp. 750–757 (↗ pp. 32, 39, 78).

[201] G. Fu, P. Corradi, A. Menciassi, and P. Dario. “An Integrated Triangulation Laser Scanner for Obstacle Detection of Miniature Mobile Robots in Indoor Environment.” In: *IEEE/ASME Transactions on Mechatronics* 16.4 (2011), pp. 778–783 (↗ p. 223).

[202] Y. Fu, S. Tully, G. Kantor, and H. Choset. “Monte Carlo Localization Using 3D Texture Maps.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2011, pp. 482–487 (↗ pp. 39, 73).

[203] Y. Fuse, M. Furuya, M. Nishimura, C. Yoshimura, H. Watanabe, T. Tanzawa, S. Kotani, and N. Kiyohiro. “Indoor Localization Using Infrared Global Vision System and LRF for Twin Brushes Floor Polishing Robots.” In: *Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA 11)*. 2011, pp. 1757–1762 (↗ p. 10).

[204] A. Gagliardo, C. Filannino, P. Iolaè, T. Pecchia, M. Wikelski, and G. Vallortigara. “Olfactory Lateralization in Homing Pigeons: a GPS Study on Birds Released With Unilateral Olfactory Inputs.” In: *The Journal of Experimental Biology* 214.4 (2011), pp. 593–598 (↗ p. 61).

[205] A. Gagliardo, P. Iolaè, M. Savini, and J. M. Wild. “Olfactory Navigation in Homing Pigeons.” In: *Annals of the New York Academy of Sciences* 1170.1 (2009), pp. 434–437 (↗ p. 61).

[206] C. Gamallo, M. Mucientes, and C. Regueiro. “Visual FastSLAM Through Omnivision.” In: *Proceedings of the Towards Autonomous Robotic Systems (TAROS 09)*. 2009, pp. 128–135 (↗ pp. 26, 39, 71).

[207] E. Gambao and M. Hernando. “Control System for a Semi-Automatic Façade Cleaning Robot.” In: *Proceedings of the International Symposium of Automation and Robotics in Construction (ISARC 06)*. 2006, pp. 406–411 (↗ p. 10).

[208] X. Gao, Y. Wang, D. Zhou, and K. Kikuchi. “Floor-Cleaning Robot Using Omni-Directional Wheels.” In: *Industrial Robot: An International Journal* 36.2 (2009), pp. 157–164 (↗ p. 10).

[209] E. Garcia, M. A. Jimenez, P. G. De Santos, and M. Armada. “The Evolution of Robotics Research.” In: *IEEE Robotics and Automation Magazine* 14.1 (2007), pp. 90–103 (↗ pp. 1, 9, 10).

[210] J. Gaspar, N. Winters, and J. Santos-Victor. “Vision-Based Navigation and Environmental Representation with an Omni-Directional Camera.” In: *IEEE Transactions on Robotics and Automation* 16.4 (2000), pp. 890–898 (↗ pp. 32, 33, 68, 72, 81–83, 170).

[211] J. Gaspar, N. Winters, E. Grossmann, and J. Santos-Victor. “Toward Robot Perception Through Omnidirectional Vision.” In: *Innovations in Intelligent Machines*. Ed. by J. Kacprzyk, J. S. Chahl, L. C. Jain, A. Mizutani, and M. Sato-Ilic. Vol. 70. Studies in Computational Intelligence (SCI). Springer, 2007, pp. 223–270 (↗ pp. 32, 33, 68, 72, 73, 81–83).

[212] B. Gates. “A Robot in Every Home.” In: *Scientific American* 296.1 (2007), pp. 58–65 (↗ p. 1).

[213] L. Gerstmayr. *Local Visual Homing with Differential Flow Methods*. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2007 (↗ pp. 17, 50, 55, 142).

[214] L. Gerstmayr, F. Röben, M. Krzykawski, S. Kreft, D. Venjakob, and R. Möller. “A Vision-Based Trajectory Controller for Autonomous Cleaning Robots.” In: *Autonome Mobile Systeme (AMS 09)*. Ed. by R. Dillmann, J. Beyerer, C. Stiller, M. Zöllner, and T. Gindele. Informatik aktuell. Springer, 2009, pp. 65–72 (↗ pp. 89, 91).

[215] L. Gerstmayr, F. Röben, and R. Möller. “From Insect Visual Homing to Autonomous Robot Cleaning.” In: *Spatial Cognition 2008, Workshop on Biologically Motivated Models of Spatial Behaviour*. 2008 (↗ pp. 89, 91).

[216] L. Gerstmayr. *Derivation of the “Koenderink-equation”*. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2006 (↗ pp. 59, 223).

[217] L. Gerstmayr. *Blockmatching Along Flow Lines*. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2007 (↗ pp. 17, 50, 59, 142, 223).

[218] L. Gerstmayr. *Performance Measures for Local Visual Homing*. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2007 (↗ p. 142).

[219] L. Gerstmayr, S. Kreft, R. Möller, and F. Röben. “Verfahren zur Navigation eines selbstfahrenden Bodenbearbeitungsgerätes.” German Patent DE102007016802B3. 2008. In German. (↗ p. 91).

[220] L. Gerstmayr-Hillen and R. Möller. *Loop-Closure Detection and Visual Compass Based on Global Image Comparisons.* Poster presented at the workshop “Insect Homing: Mechanisms and Models”, Bielefeld, Germany. 2011 (↗ p. 117).

[221] L. Gerstmayr-Hillen, O. Schlüter, M. Krzykawski, and R. Möller. *Loop-Closure Detection Based on Global Image Signatures.* Poster presented at the workshop “Insect Homing: Mechanisms and Models”, Bielefeld, Germany. 2011 (↗ p. 169).

[222] L. Gerstmayr-Hillen, F. Röben, M. Krzykawski, S. Kreft, D. Venjakob, and R. Möller. “Dense Topological Maps and Partial Pose Estimation for Visual Control of an Autonomous Cleaning Robot.” In: *Robotics and Autonomous Systems* 61.5 (May 2013), pp. 497–516 (↗ pp. 9, 17, 27, 47, 62, 89, 91).

[223] L. Gerstmayr-Hillen, O. Schlüter, M. Krzykawski, and R. Möller. “Parsimonious Loop-Closure Detection Based on Global Image-Descriptors of Panoramic Images.” In: *Proceedings of the IEEE International Conference on Advanced Robotics (ICAR 2011).* 2011, pp. 576–581 (↗ pp. 32, 89, 117, 133, 169).

[224] C. Geyer and K. Daniilidis. “A Unifying Theory for Central Panoramic Systems and Practical Implications.” In: *Proceedings of the European Conference on Computer Vision (ECCV 00).* Ed. by D. Vernon. Vol. 1843. Lecture Notes in Computer Science (LNCS). Springer, 2000, pp. 445–461 (↗ p. 24).

[225] A. Giachetti. “Matching Techniques to Compute Image Motion.” In: *Image and Vision Computing* 18.3 (2000), pp. 247–260 (↗ pp. 22, 30, 33, 42, 45, 59, 117, 119, 123, 124, 218).

[226] A. Gil, O. M. Mozos, M. Ballesta, and O. Reinoso. “A Comparative Evaluation of Interest Point Detectors and Local Descriptors for Visual SLAM.” In: *Machine Vision Applications* 21.6 (2010), pp. 905–920 (↗ pp. 33, 169).

[227] S. Gillner, A. M. Weiss, and H. A. Mallot. “Visual Homing in the Abscence of Feature-Based Landmark Information.” In: *Cognition* 109 (2008), pp. 105–122 (↗ pp. 43, 61).

[228] C. Giovannangeli and P. Gaussier. “Autonomous Vision-Based Navigation. Goal-oriented action planning by transient states prediction, cognitive map building, and sensory-motor learning.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 08).* 2008, pp. 676–683 (↗ pp. 83, 86).

[229] C. Giovannangeli and P. Gaussier. “Interactive Teaching for Vision-Based Mobile Robots: A Sensory-Motor Approach.” In: *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans* 40.1 (2010), pp. 13–28 (↗ pp. 83, 86).

[230] C. Giovannangeli, P. Gaussier, and G. Desilles. “Robust Mapless Outdoor Vision-Based Navigation.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 06).* 2006, pp. 3293–3300 (↗ pp. 83, 86).

[231] C. Giovannangeli, P. Gaussier, and B. J. P. “Robustness of Visual Place Cells in Dynamic Indoor and Outdoor Environment.” In: *International Journal of Advanced Robotic Systems* 3.2 (2006), pp. 115–124 (↗ pp. 83, 86).

[232] M. Giurfa. “Cognition with Few Neurons: Higher-Order Learning in Insects.” In: *Trends in Neuroscience* 35.5 (2013), pp. 285–294 (↗ p. 42).

[233] J. M. Gluckman. “Higher Order Whitening of Natural Images.” In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 05).* 2005, pp. 354–360 (↗ p. 122).

[234] J. Gluckman, S. K. Nayar, and K. J. Thoresz. “Real-Time Omnidirectional and Panoramic Stereo.” In: *Proceedings of the DARPA Image Understanding Workshop.* 1998, pp. 299–303 (↗ p. 26).

[235] T. Goedemé, T. Tuytelaars, L. Van Gool, G. Vanacker, and M. Nuttin. “Feature Based Omnidirectional Sparse Visual Path Following.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 05).* 2005, pp. 1806–1811 (↗ pp. 26, 32, 36, 57, 68, 71, 72, 82, 83, 89).

[236] T. Goedemé, T. Tuytelaars, L. Van Gool, G. Vanacker, and M. Nuttin. “Omnidirectional Sparse Visual Path Following with Occlusion-Robust Feature Tracking.” In: *Proceedings of the Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras (OMNIVIS 05).* 2005 (↗ pp. 32, 35, 36, 57, 68, 72, 82, 83, 89).

[237] T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool. “Omnidirectional Vision Based Topological Navigation.” In: *International Journal of Computer Vision* 74.3 (2007), pp. 219–236 (↗ pp. 26, 32, 35, 36, 39, 57, 68, 70–72, 81–83, 89).

[238] A. Goldhoorn, A. Ramisa, R. López de Mántaras, and R. Toledo. “Using the Average Landmark Vector Method for Robot Homing.” In: *Proceedings of the Conference on Artificial Intelligence Research and Development.* 2007, pp. 331–338 (↗ p. 54).

[239] M. Golfarelli, D. Maio, and S. Rizzi. “Elastic Correction of Dead-Reckoning Errors in Map Building.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 98).* 1998, pp. 905–911 (↗ p. 39).

[240] M. Golfarelli, D. Maio, and S. Rizzi. “Correction of Dead-Reckoning Errors in Map Building for Mobile Robots.” In: *IEEE Transactions on Robotics and Automation* 17.1 (2001), pp. 37–47 (↗ p. 39).

[241] R. C. González and R. E. Woods. *Digital Image Processing.* 3rd ed. Prentice Hall, 2008 (↗ p. 123).

[242] J.-J. Gonzalez-Barbosa and S. Lacroix. "Rover Localization in Natural Environments by Indexing Panoramic Images." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 02).* 2002, pp. 1365–1370 (↗ pp. 29, 31, 32, 82, 129, 170, 172, 179).

[243] G. Götze. "Adaptive Kamerasteuerung zur Erhöhung der Toleranz gegen Beleuchtungsänderungen." Bachelor's Thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2008. In German. (↗ pp. 17, 21).

[244] S. Gourichon, J.-A. Meyer, and P. Pirim. "Using Coloured Snapshots for Short-Range Guidance in Mobile Robots." In: *International Journal of Robotics and Automation* 17.4 (2002), pp. 154–162 (↗ p. 54).

[245] P. Graham, A. Philippides, and B. Baddeley. "Animal Cognition: Multi-Modal Interactions in Ant Learning." In: *Current Biology* 20.15 (2010), R639–R640 (↗ pp. 61, 83, 86).

[246] P. Graham. "Insect Navigation." In: *Encyclopedia of Insect Navigation.* Ed. by M. D. Breed and J. Moore. Vol. 2. Oxford Academic Press, pp. 167–175 (↗ p. 60).

[247] P. Graham and K. Cheng. "Ants Use the Panoramic Skyline as a Visual Cue During Navigation." In: *Current Biology* 19.20 (2009), R935–R937 (↗ pp. 42, 60).

[248] P. Graham and K. Cheng. "Which Portion of the Natural Panorama is Used for View-Based Navigation in the Australian Desert Ant?" In: *Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology* 195.7 (2009), pp. 681–689 (↗ pp. 42, 60, 212).

[249] P. Graham and T. S. Collett. "Bi-Directional Route Learning in Wood Ants." In: *Journal of Experimental Biology* 209.18 (2006), pp. 3677–3684 (↗ p. 86).

[250] P. Graham and A. Philippides. "Insect-Inspired Vision and Visually Guided Control." In: *Encyclopedia of Nanotechnology.* Ed. by B. Bushan. Springer, 2012, pp. 1122–1127 (↗ pp. 56, 86).

[251] P. Greguss. "Panoramic Imaging Block for Three-Dimensional Space." US 4,566,763. 1986 (↗ pp. 26, 104).

[252] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. "A Tutorial on Graph-Based SLAM." In: *Intelligent Transportation Systems Magazine* 2.4 (2010), pp. 31–43 (↗ pp. 39, 40).

[253] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. "A Tree Parameterization for Efficiently Computing Maximum Likelihood Maps Using Gradient Descent." In: *Proceedings of Robotics: Science and Systems (RSS 07).* 2007, P09 (↗ p. 39).

[254] G. Grisetti, C. Stachniss, and W. Burgard. "Nonlinear Constraint Network Optimization for Efficient Map Learning." In: *IEEE Transactions on Intelligent Transportation Systems* 10.3 (2009), pp. 428–439 (↗ p. 39).

[255] H.-M. Gross and H.-J. Böhme. "PERSES, a Vision-Based Interactive Mobile Shopping Assistant." In: *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics.* 2000, pp. 80–85 (↗ pp. 78, 89, 213).

[256] H.-M. Gross, A. König, H.-J. Böhme, and C. Schröter. "Vision-Based Monte Carlo Self-Localization for a Mobile Service Robot Acting as Shopping Assistant in a Home Store." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems.* 2002, pp. 256–262 (↗ pp. 32, 46, 78, 89, 213).

[257] H.-M. Gross, A. König, and S. Müller. "Omniview-Based Concurrent Map Building and Localization Using Adaptive Appearance Maps." In: *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics.* 2005, pp. 3510–3515 (↗ pp. 32, 46, 78, 89, 213).

[258] H.-M. Gross, A. König, C. Schröter, and H.-J. Böhme. "Omnivision-Based Probabilistic Self-Localization for a Mobile Shopping Assistant Continued." In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 03).* 2003, pp. 1505–1511 (↗ pp. 78, 89, 213).

[259] M. Grundland and N. Dodgson. "Automatic Contrast Enhancement by Histogram Warping." In: *Computer Vision and Graphics. Proceedings of the International Conference on Computer Vision and Graphics (ICCVG 04).* Ed. by K. Wojciechowski, B. Smolka, H. Palus, R. S. Kozera, W. Skarbek, and L. Noakes. Vol. 32. Computational Imaging and Vision. Springer, 2006, pp. 293–300 (↗ pp. 167, 218).

[260] M. Grundland and N. Dodgson. "Interactive Contrast Enhancement by Histogram Warping." In: *Computer Vision and Graphics. Proceedings of the International Conference on Computer Vision and Graphics (ICCVG 04).* Ed. by K. Wojciechowski, B. Smolka, H. Palus, R. S. Kozera, W. Skarbek, and L. Noakes. Vol. 32. Computational Imaging and Vision. Springer, 2006, pp. 832–838 (↗ pp. 167, 218).

[261] S. Gu and Q. Chen. "Building Topological Map Using Omnidirectional Image Matching." In: *Proceedings of the International Conference on System Science and Engineering (ICSSE 11).* 2011, pp. 282–287 (↗ pp. 32, 46, 77, 211).

[262] V. C. Guizilini and J. Okamoto. "Solving the Online SLAM Problem with an Omnidirectional Vision System." In: *Proceedings of the International Conference on Advances in Neural Information Processing (ICONIP 09).* 2009, pp. 1110–1117 (↗ pp. 39, 71).

[263] J.-S. Gutmann, G. Brisson, E. Eade, P. Fong, and M. Munich. "Vector Field SLAM." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 10).* 2010, pp. 236–242 (↗ pp. 15, 16).

[264] J.-S. Gutmann, E. Eade, P. Fong, and M. Munich. "A Constant-Time Algorithm for Vector Field SLAM Using an Exactly Sparse Extended Information Filter." In: *Proceedings of Robotics: Science and Systems (RSS 10).* MIT Press, 2010, P25 (↗ pp. 15, 16).

[265] H. Hadj Abdelkader, Y. Mezouar, P. Martinet, and F. Chaumette. “Catadioptric Visual Servoing from 3D Straight Lines.” In: *IEEE Transactions on Robotics* 24.3 (2008), pp. 652–665 (↗ pp. 35, 49, 57).

[266] V. V. Hafner. “Learning Places in Newly Explored Environments.” In: *Proceedings of the International Conference on the Simulation of Adaptive Behavior (SAB 00)*. 2000, pp. 111–120 (↗ pp. 39, 83, 86).

[267] V. V. Hafner. “Adaptive Homing. Robotic Exploration Tours.” In: *Adaptive Behavior* 9.3-4 (Jan. 2001), pp. 131–141 (↗ p. 54).

[268] V. V. Hafner. “Cognitive Maps in Rats and Robots.” In: *Adaptive Behavior* 13.2 (2005), pp. 87–96 (↗ pp. 77, 83, 86).

[269] V. V. Hafner. “Robots as Tools for Modelling Navigation Skills — A Neural Cognitive Map Approach.” In: *Robotics and Cognitive Approaches to Spatial Mapping*. Ed. by M. Jefferies and W.-K. Yeap. Vol. 38. Springer Tracts in Advanced Robotics (STAR). Springer, 2008, pp. 315–324 (↗ pp. 77, 83, 86).

[270] M. Hägele, K. Nilsson, and J. N. Pires. “Industrial Robots.” In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 1253–1281 (↗ p. 1).

[271] J. Ham, Y. Lin, and D. D. Lee. “Learning Nonlinear Appearance Manifolds for Robot Localization.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 05)*. 2005, pp. 2971–2976 (↗ pp. 3, 78).

[272] D. A. Hamilton, K. G. Akers, M. P. Weisend, and R. J. Sutherland. “How Do Room and Apparatus Cues Control Navigation in the Morris Water Task?” In: *Journal of Experimental Psychology: Animal Behavior Processes* 33.2 (2007), pp. 100–114 (↗ pp. 43, 61).

[273] D. A. Hamilton, J. P. R. F. T. C. K. G. Akers T. E. Johnson, R. J. Sutherland, M. P. Weisend, and E. S. Redhead. “The Relative Influence of Place and Direction in the Morris Water Task.” In: *Journal of Experimental Psychology: Animal Behavior Processes* 34.1 (2008), pp. 31–53 (↗ pp. 43, 61).

[274] B. Hammer and T. Villmann. “Generalized Relevance Learning Vector Quantization.” In: *Neural Networks* 15.8-9 (2002), pp. 1059–1068 (↗ p. 212).

[275] D. J. Hand and R. J. Till. “A Simple Generalization of the Area Under the ROC Curve for Multiple Class Classification Problems.” In: *Machine Learning* 45.2 (2001), pp. 171–186 (↗ pp. 139, 182).

[276] T. Hansen and H. Neumann. “A Simple Cell Model with Dominating Opponent Inhibition for Robust Image Processing.” In: *Neural Networks* 17.5-6 (2004), pp. 647–662 (↗ p. 122).

[277] C. Harris and M. Stephens. “A Combined Corner and Edge Detection.” In: *Proceedings of the Fourth Alvey Vision Conference*. 1988, pp. 147–151 (↗ pp. 15, 35, 59).

[278] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003 (↗ pp. 24, 26, 28, 36, 39, 57, 65, 106, 107).

[279] M. E. Hasselmo. “Models of Hippocampus.” In: *Scholarpedia* 6.5 (2011), p. 1371 (↗ pp. 66, 86).

[280] H. Haussecker and H. Spies. “Motion.” In: *Handbook of Computer Vision and Applications*. Ed. by B. Jähne, H. Haussecker, and P. Geissler. Vol. 2. Academic Press, 1999, pp. 309–396 (↗ pp. 28, 54, 219, 223).

[281] Q. He. “Design of Central Air Conditioning Duct Cleaning Robot Control System.” In: *Mechanical and Electrical Engineering Magazine* 28.8 (2011), pp. 944–947 (↗ p. 10).

[282] P. Heinemann, J. Haase, and A. Zell. “A Combined Monte-Carlo Localization and Tracking Algorithm for Robocup.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 06)*. 2006, pp. 1535–1540 (↗ pp. 39, 75, 89).

[283] J. Hertzberg, T. Christaller, F. Kirchner, U. Licht, and E. Rome. “Sewer Robotics.” In: *From Animals to Animats. Proceedings of the International Conference on Simulation of Adaptive Behavior (SAB 98)*. MIT Press, 1998, pp. 427–436 (↗ p. 10).

[284] R. W. Hicks and E. L. Hall. “Survey of Robot Lawn Mowers.” In: *Intelligent Robots and Computer Vision XIX: Algorithms, Techniques, and Active Vision*. Ed. by D. P. Casasent. Vol. 4197. Proceedings of SPIE. 2000, pp. 262–269 (↗ p. 10).

[285] H. Hirschmüller. “Stereo Processing by Semiglobal Matching and Mutual Information.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30.2 (2008), pp. 328–341 (↗ p. 128).

[286] H. Hirschmüller and D. Scharstein. “Evaluation of Cost Functions for Stereo Matching.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 07)*. 2007 (↗ pp. 128, 155).

[287] K. A. Hobson and D. Ryan Norris. “Animal Migration: A Context for Using New Techniques and Approaches.” In: *Tracking Animal Migration with Stable Isotopes*. Ed. by K. A. Hobson and L. I. Wassenaar. Vol. 2. Elsevier, 2008, pp. 1–19 (↗ p. 61).

[288] U. Homberg. “In Search of the Sky Compass in the Insect Brain.” In: *Naturwissenschaften* 91.5 (2004), pp. 199–208 (↗ p. 47).

[289] M. Horst. *Visual Distance Measurement*. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2011 (↗ p. 223).

[290] G. Hu, Z. Hu, and H. Wang. “Complete Coverage Path Planning for Road Cleaning Robot.” In: *Proceedings of the IEEE International Conference on Networking, Sensing and Control (ICNSC 10)*. 2010, pp. 643–648 (↗ p. 10).

[291] F. Huang, R. Klette, and K. Scheible. “Epipolar Geometry.” In: *Panoramic Imaging. Sensor-Line Cameras and Laser Range Finders*. Wiley, 2008, pp. 81–98 (↗ p. 26).

[292] F.-S. Huang and K.-T. Song. “Vision SLAM Using Omni-Directional Visual Scan Matching.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 08)*. 2008, pp. 1588–1593 (↗ pp. 26, 35, 39, 71).

[293] Y. Huang and X. H. Zhuang. “Motion-Partitioned Adaptive Block Matching for Video Compression.” In: *Proceedings of the IEEE International Conference on Image Processing (ICIP 95)*. 1995, pp. 554–557 (↗ p. 58).

[294] W. Hübner and H. A. Mallot. “Metric Embedding of View Graphs: A Vision and Odometry-Based Approach to Cognitive Mapping.” In: *Autonomous Robots* 23.3 (2007), pp. 183–196 (↗ pp. 39, 77).

[295] B. Huhle, T. Schairer, A. Schilling, and W. Strasser. “Learning to Localize with Gaussian Process Regression on Omnidirectional Image Data.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 10)*. Oct. 2010, pp. 5208–5213 (↗ pp. 32, 78).

[296] B. Huhle, T. Schairer, and W. Strasser. “Normalized Cross-Correlation Using SOFT.” In: *Proceedings of the International Workshop on Local and Non-Local Approximation in Image Processing*. Aug. 2009, pp. 82–86 (↗ p. 45).

[297] F. v. Hundelshausen, S. Behnke, and R. Rojas. “An Omnidirectional Vision System That Finds and Tracks Color Edges and Blobs.” In: *Robocup 2001: Robot Soccer World Cup V*. Ed. by R. P. S. Noske, S. Coradeschi, and S. Tadokoro. Vol. 2377. Lecture Notes in Artifical Intelligence (LNAI). Springer, 2002, pp. 374–379 (↗ pp. 75, 89).

[298] T. A. Hurly, S. Franz, and S. D. Healy. “Do Rufous Hummingbirds (Selasphorus rufus) Use Visual Beacons?” In: *Animal Cognition* 13.2 (2010), pp. 377–383 (↗ p. 60).

[299] S. Hutchinson, G. Hager, and P. Corke. “A Tutorial on Visual Servo Control.” In: *IEEE Transactions on Robotics and Control* 12.5 (1996), pp. 651–670 (↗ p. 49).

[300] IFR Statistical Department. *World Robotics 2012. Service Robots.* International Federation of Robotics, Frankfurt am Main, Germany, 2012 (↗ p. 1).

[301] S. Ikeda and J. Miura. “3D Indoor Environment Modeling by a Mobile Robot with Omnidirectional Stereo and Laser Range Finder.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 06)*. 2006, pp. 3435–3440 (↗ pp. 73, 74).

[302] A. Imiya, A. Torii, and H. Sugaya. “Optical Flow Computations of Omni-Directional Images.” In: *Imaging Beyond the Pinhole Camera*. Ed. by K. Daniilidis and R. Klette. Vol. 33. Computational Imaging. Springer, 2006, pp. 143–162 (↗ p. 29).

[303] A. Isidori. “Control Theory for Automation: Fundamentals.” In: *Handbook of Automation*. Ed. by S. Y. Nof. Springer, 2009, pp. 147–172 (↗ pp. 49, 58).

[304] L. Itti. “Visual Salience.” In: *Scholarpedia* 2.9 (2007), p. 3327 (↗ p. 43).

[305] B. Jähne. *Digital Image Processing*. 6th ed. Springer, 2005 (↗ pp. 28, 54, 55, 219, 223, 224).

[306] M. Jamzad, A. H. Khodabakshi, and V. S. Mirrokni. “An Omnidirectional Vision System for Localization and Object Detection in Middle Size Robocup.” In: *Proceeding of the IAESTED International Conference on Applied Modelling and Simulation (AMS 02)*. 2002 (↗ pp. 75, 89).

[307] G. Jang, S. Kim, and I. Kweon. “Single Camera Catadioptric Stereo System.” In: *Proceedings of the Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Camera (OMNIVIS 05)*. 2005 (↗ p. 26).

[308] M. E. Jefferies and W. K. Yeap. *Robot and Cognitive Approaches to Spatial Mapping*. Vol. 38. Springer Tracts in Advanced Robotics (STAR). Springer, 2008 (↗ p. 66).

[309] W. Y. Jeong and K. M. Lee. “CV-SLAM: A New Ceiling Vision-Based SLAM Technique.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2005, pp. 3195–3200 (↗ pp. 15, 40).

[310] W. Y. Jeong and K. M. Lee. “Visual SLAM with Line and Corner Features.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2006, pp. 2570–2575 (↗ pp. 15, 40).

[311] M. Jogan and A. Leonardis. “Robust Localization Using Panoramic View-Based Recognition.” In: *Proceedings of the International Conference on Pattern Recognition (ICPR 00)*. 2000, pp. 136–139 (↗ pp. 32, 33, 82, 170).

[312] M. Jogan and A. Leonardis. “Robust Localization Using Panoramic View-Based Recognition.” In: *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR 00)*. 2000, pp. 136–139 (↗ pp. 32, 33, 82).

[313] M. Jogan and A. Leonardis. “Robust Localization Using an Omnidirectional Appearance-Based Subspace Model of the Environment.” In: *Robotics and Autonomous Systems* 45.1 (2003), pp. 51–72 (↗ pp. 32, 33, 81, 82, 170).

[314] J. Jong. “Kalmanfilter zur Positionsbestimmung auf mäandrierenden Bahnen.” Diploma Thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2008. In German. (↗ pp. 79, 113, 214).

[315] S. P. D. Judd and T. S. Collett. “Multiple Stored Views and Landmark Guidance in Ants.” In: *Nature* 392.6677 (1998), pp. 710–714 (↗ p. 86).

[316] M.-J. Jung, H. Myung, S.-G. Hong, D.-R. Park, H.-K. Lee, and S. Bang. “Structured Light 2D Range Finder for Simultaneous Localization and Map-Building (SLAM) in Home Environments.” In: *Proceedings of the International Symposium on Micro-Nanomechatronics and Human Science*. 2004, pp. 371–376 (↗ p. 223).

[317] M.-J. Jung, H. Myung, H.-K. Lee, and S. Bang. “Ambiguity Resolving in Structured Light 2D Range Finder for SLAM Operation for Home Robot Applications.” In: *Proceedings of the IEEE Workshop on Advanced Robotics and its Social Impacts*. 2005, pp. 18–23 (↗ p. 223).

[318] M. Kaess and F. Dellaert. “Probabilistic Structure Matching for Visual SLAM with a Multi-Camera Rig.” In: *Computer Vision and Image Understanding* 114.2 (2010), pp. 286–296 (↗ p. 26).

[319] T. Kailath. “The Divergence and Bhattacharyya Distance Measures in Signal Selection.” In: *IEEE Transactions on Communication Technology* 15.1 (1967) (↗ p. 176).

[320] E. R. Kandel, J. H. Schwartz, and T. M. Jessell. *Principles of Neural Science*. McGraw-Hill, 2000 (↗ p. 122).

[321] Y. Kaneta, Y. Hagisaka, and K. Ito. “Determination of Time to Contact and Application to Timing Control of Mobile Robot.” In: *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO 10)*. 2010, pp. 161–166 (↗ p. 29).

[322] R. Kawanishi, A. Yamashita, and T. Kaneko. “Estimation of Camera Motion with Feature Flow Model for 3D Environment Modeling by Using Omni-Directional Camera.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 09)*. 2009, pp. 3089–3094 (↗ pp. 71–73).

[323] R. Kawanishi, A. Yamashita, and T. Kaneko. “Construction of 3D Environment Model from an Omni-Directional Image Sequence.” In: *Proceedings of the Asian International Symposium on Mechatronics (AISM 08*. 2008, pp. 1–6 (↗ pp. 71–73).

[324] R. Kawanishi, A. Yamashita, and T. Kaneko. “Three-Dimensional Environment Model Construction from an Omnidirectional Image Sequence.” In: *Journal of Robotics and Mechatronics* 21.5 (2009), pp. 574–582 (↗ pp. 71–73).

[325] D. M. Kelly, S. Kippenbrock, J. Templeton, and A. C. Kamil. “Use of a Geometric Rule or Absolute Vectors: Landmark Use by Clark’s Nutcrackers (Nucifraga columbiana).” In: *Brain Research Bulletin* 76.3 (2008), pp. 293–299 (↗ p. 60).

[326] D. M. Kelly, A. C. Kamil, and K. Cheng. “Landmark Use by Clark’s Nutcrackers (Nucifraga columbiana): Influence of Disorientation and Cue Rotation on Distance and Direction Estimates.” In: *Animal Cognition* 13.1 (2010), pp. 175–188 (↗ p. 60).

[327] J. Kessler, A. König, and H.-M. Gross. “An Improved Sensor Model on Appearance Based SLAM.” In: *Autonome Mobile Systeme (AMS 09)*. Ed. by R. Dillmann, J. Beyerer, C. Stiller, J. M. Zöllner, and T. Gindele. Informatik aktuell. Springer, 2009, pp. 153–160 (↗ pp. 21, 35, 78, 79, 89, 213).

[328] M. Kesting. “Verbesserung des Blockmatching-Verfahrens durch Selektion und Tracking von Features.” Bachelor’s Thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2008. In German. (↗ pp. 17, 50, 59).

[329] C. Y. Kim, J.-S. Kim, and K. S. Hong. “Real-Time Simultaneous Localization and Mapping Using Omnidirectional Vision.” In: *Proceedings of the International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 07)*. 2007, pp. 474–479 (↗ pp. 35, 39, 71).

[330] J. Kim, S. Woo, J. Kim, J. Do, S. Kim, and S. Bae. “Inertial Navigation System for an Automatic Guided Vehicle with Mecanum Wheels.” In: *International Journal of Precision Engineering and Manufacturing* 13.3 (2012), pp. 379–386 (↗ p. 10).

[331] J.-H. Kim and M. J. Chung. “SLAM with Omni-Directional Stereo Vision Sensor.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 03)*. 2003, pp. 442–447 (↗ pp. 39, 40, 71).

[332] J. Kim, K.-J. Yoon, J.-S. Kim, and I. Kweon. “Visual SLAM by Single-Camera Catadioptric Stereo.” In: *Proceedings of the SICE-ICASE International Joint Conference (SICE-ICASE 06)*. 2006, pp. 2005–2009 (↗ pp. 40, 71).

[333] P. Kim, M. D. Szenher, and B. Webb. “Entropy-Based Visual Homing.” In: *Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA 09)*. Aug. 2009, pp. 3601–3606 (↗ p. 52).

[334] J. J. Koenderink and A. J. van Doorn. “Representation of Local Geometry in the Visual System.” In: *Biological Cybernetics* 55.6 (1987), pp. 367–375 (↗ pp. 59, 223).

[335] K. C. Koh, H. J. Choi, J. S. Kim, K. W. Ko, and H. S. Cho. “Sensor-Based Navigation of Air-Duct Inspection Mobile Robots.” In: *Optomechatronic Systems*. Ed. by G. K. K. Hyungsuck Cho. Vol. 4190. Proceedings of SPIE. 2001, pp. 202–211 (↗ p. 10).

[336] M. Kohler and R. Wehner. “Idiosyncratic Route-Based Memories in Desert Ants, Melophorus Bagoti: How Do They Interact with Path-Integration Vectors?” In: *Neurobiology of Learning and Memory* 83.1 (2005), pp. 1–12 (↗ p. 86).

[337] T. Kollmeier, F. Röben, W. Schenck, and R. Möller. “Spectral Contrasts for Landmark Navigation.” In: *Journal of the Optical Society of America A* 24.1 (2007), pp. 1–10 (↗ pp. 60, 217).

[338] A. König, J. Kessler, and H.-M. Gross. “A Graph Matching Technique for an Appearance-Based, Visual SLAM-Approach Using Rao-Blackwellized Particle Filters.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 08)*. 2008, pp. 1576–1581 (↗ pp. 32, 78, 89, 213).

[339] A. König, J. Kessler, and H.-M. Gross. “Improvements for an Appearance-Based SLAM-Approach for Large-Scale Environments.” In: *Proceedings of the European Conference on Mobile Robotics (ECMR 09)*. 2009, pp. 235–240 (↗ pp. 35, 78, 79, 87, 89, 213).

[340] A. König, S. Müller, and H.-M. Gross. “Appearance-Based CML Approach for a Home Store Environment.” In: *Proceedings of the European Conference on Mobile Robotics (ECMR 05)*. 2005, pp. 206–211 (↗ pp. 32, 46, 78, 89, 213).

[341] D. Kragic and M. Vincze. “Vision for Robotics.” In: *Foundations and Trends in Robotics* 1.1 (2010), pp. 1–78 (↗ pp. 17, 40).

[342] S. Kreft. “Reinigungstrajektorien mobiler Roboter unter visueller Steuerung.” Diploma Thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2007. In German. (↗ pp. 21, 88, 91, 97, 231).

[343] B. Kröse, O. Booij, and Z. Zivkovic. “A Geometrically Constrained Image Similarity Measure for Visual Mapping, Localization and Navigation.” In: *Proceedings of the European Conference on Mobile Robots (ECMR 07)*. 2007, pp. 168–174 (↗ pp. 36, 68, 81, 88).

[344] B. Kröse, R. Bunschoten, S. T. Hagen, B. Terwijn, and N. Vlassis. “Household Robots Look and Learn: Environment Modeling and Localization from an Omnidirectional Vision System.” In: *IEEE Robotics and Automation Magazine* 11.4 (2004), pp. 45–52 (↗ p. 74).

[345] M. Krzykawski. *Model-Free Calibration Method for a Generic SVP Omnidirectional Camera*. Technical report. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2011 (↗ pp. 104, 201).

[346] B. Kuipers. “The Spatial Semantic Hierarchy.” In: *Artificial Intelligence* 119.1-2 (2000), pp. 191–233 (↗ p. 79).

[347] B. Kuipers. “An Intellectual History of the Spatial Semantic Hierarchy.” In: *Robot and Cognitive Approaches to Spatial Mapping*. Ed. by M. E. Jefferies and W. K. Yeap. Vol. 38. Springer Tracts in Advanced Robotics (STAR). Springer, 2008, pp. 243–246 (↗ p. 79).

[348] S. Kullback. *Information Theory and Statistics*. Wiley, 1959 (↗ p. 176).

[349] R. Kümmerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss, and A. Kleiner. “On Measuring the Accuracy of SLAM Algorithms.” In: *Autonomous Robots* 27.4 (2009), pp. 387–407 (↗ p. 88).

[350] T. B. Kwon, J. B. Song, and S. C. Kang. “MCL-Based Global Localization of Cleaning Robot Using Fast Rotation-Invariant Corner Matching Method.” In: *Proceedings of the International Conference on Control, Automation, and Systems (ICROS 10)*. 2010, pp. 1988–1992 (↗ p. 15).

[351] O. Labbani-Igbida, C. Charron, and E. Mouaddib. “Extraction of Haar Integral Features on Omnidirectional Images.” In: *Pattern Recognition*. Ed. by K. Franke, K.-R. Müller, B. Nickolay, and R. Schäfer. Vol. 4174. Lecture Notes in Computer Science (LNCS). Springer, 2006, pp. 334–343 (↗ p. 32).

[352] O. Labbani-Igbida, C. Charron, and E. Mouaddib. “Haar Invariant Signatures and Spatial Recognition Using Omnidirectional Visual Information Only.” In: *Autonomous Robots* 30.3 (2011), pp. 333–349 (↗ pp. 32, 82, 139, 170).

[353] F. Labrosse. “Visual Compass.” In: *Proceedings of the Towards Autonomous Robotic Systems (TAROS 04)*. 2004, pp. 85–92 (↗ p. 45).

[354] F. Labrosse. “The Visual Compass: Performance and Limitations of an Appearance-Based Method.” In: *Journal of Field Robotics* 23.10 (2006), pp. 913–941 (↗ pp. 45, 52).

[355] F. Labrosse. “Short and Long-Range Visual Navigation Using Warped Panoramic Images.” In: *Robotics and Autonomous Systems* 55.9 (2007), pp. 675–684 (↗ p. 82).

[356] D. Lambrinos, R. Möller, T. Labhart, R. Pfeifer, and R. Wehner. “A Mobile Robot Employing Insect Strategies for Navigation.” In: *Robotics and Autonomous Systems* 30.1 (2000), pp. 39–64 (↗ pp. 53, 54).

[357] T. Langlotz, D. Wagner, A. Mulloni, and D. Schmalstieg. “Online Creation of Panoramic Augmented Reality Annotations on Mobile Phones.” In: *IEEE Pervasive Computing* 11.2 (2012), pp. 56–63 (↗ p. 85).

[358] P. E. Latham and Y. Roudi. “Mutual Information.” In: *Scholarpedia* 4.1 (2009), p. 1658 (↗ p. 127).

[359] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006 (↗ p. 75).

[360] H. S. Lee and K. M. Lee. “Multi-Robot SLAM Using Ceiling Vision.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2009, pp. 912–917 (↗ pp. 15, 40).

[361] H. S. Lee and K. M. Lee. “Multiswarm Particle Filter for Vision-Based SLAM.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2009, pp. 924–929 (↗ pp. 15, 40).

[362] T. Lee, S. Baek, and S. Oh. “Sector-Based Maximal Online Coverage of Unknown Environments for Cleaning Robots with Limited Sensing.” In: *Robotics and Autonomous Systems* 59.10 (2011), pp. 698–710 (↗ p. 15).

[363] D. L. Lehner, A. G. Richter, D. R. Matthys, and J. A. Gilbert. “Characterization of the Panoramic Annular Lens.” In: *Experimental Mechanics* 36.4 (1996), pp. 333–338 (↗ p. 26).

[364] T. Lemaire and S. Lacroix. “SLAM with Panoramic Vision.” In: *Journal of Field Robotics* 24.1-2 (2007), pp. 91–111 (↗ pp. 26, 35, 39, 40, 69, 71).

[365] S. Leutenegger, M. Chli, and R. Y. Siegwart. “BRISK: Binary Robust Invariant Scalable Keypoints.” In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV 11)*. 2011, pp. 2548–2555 (↗ p. 41).

[366] A. Levin and R. Szeliski. “Visual Odometry and Map Correlation.” In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 04)*. 2004, pp. 611–618 (↗ pp. 39, 69, 77).

[367] T. S. Levitt and D. T. Lawton. “Qualitative Navigation for Mobile Robots.” In: 44.3 (1990), pp. 305–360 (↗ p. 19).

[368] M. Lhuillier. “Effective and Generic Structure from Motion Using Angular Error.” In: *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR 06)*. 2006, pp. 67–70 (↗ p. 74).

[369] M. Lhuillier. “Toward Flexible 3D Modeling Using a Catadioptric Camera.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 07)*. 2007, pp. 1–8 (↗ pp. 73, 74).

[370] M. Lhuillier. “Automatic Structure and Motion Using a Catadioptric Camera.” In: *Proceedings of the Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras (OMNIVIS 05)*. 2005 (↗ p. 74).

[371] M. Lhuillier. “Automatic Scene Structure and Camera Motion Using a Catadioptric System.” In: *Computer Vision and Image Understanding* 109.2 (2008), pp. 186–203 (↗ pp. 73, 74).

[372] J. Li and N. M. Allinson. “A Comprehensive Review of Current Local Features for Computer Vision.” In: *Neurocomputing* 71 (2008), pp. 1771–1787 (↗ pp. 33, 169).

[373] M. Lihoreau, N. E. Raine, A. M. Reynolds, R. J. Stelzer, K. S. Lim, A. D. Smith, J. L. Osborne, and L. Chittka. “Radar Tracking and Motion-Sensitive Cameras on Flowers Reveal the Development of Pollinator Multi-Destination Routes Over Large Spatial Scales.” In: *PLoS Biology* 10.9 (2012), e1001392 (↗ p. 86).

[374] J. Lim and N. Barnes. *Robust Visual Homing with Landmark Angles*. 2009 (↗ p. 58).

[375] J. Lim and N. Barnes. “Estimation of the Epipole Using Optic Flow at Antipodal Points.” In: *Computer Vision and Image Understanding* 114.2 (2010), pp. 245–253 (↗ p. 59).

[376] J. Lim, N. Barnes, and H. Li. “Estimating Relative Camera Motion from the Antipodal-Epipolar Constraint.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.10 (2010), pp. 1907–1914 (↗ p. 59).

[377] F. Linåker and M. Ishikawa. “Real-Time Appearance-Based Monte Carlo Localization.” In: *Robotics and Autonomous Systems* 54.3 (2006), pp. 205–220 (↗ pp. 3, 78).

[378] M. Liu, C. Pradalier, F. Pomerleau, and R. Siegwart. “Scale-Only Visual Homing from an Omnidirectional Camera.” In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2012, pp. 3944–3949 (↗ pp. 35, 58).

[379] M. Liu, C. Pradalier, F. Pomerleau, and R. Siegwart. “The Role of Homing in Visual Topological Navigation.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 12)*. 2012, pp. 567–572 (↗ pp. 35, 36, 58, 82, 83, 89, 114).

[380] M. Liu, D. Scaramuzza, C. Pradalier, R. Siegwart, and Q. Chen. “Scene Recognition with Omnidirectional Vision for Topological Map Using Lightweight Adaptive Descriptors.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 09)*. 2009, pp. 116–121 (↗ pp. 32, 81).

[381] M. Liu and R. Siegwart. “DP-FACT: Towards Topological Mapping and Scene Recognition with Color for Omnidirectional Camera.” In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2012, pp. 3503–3508 (↗ p. 81).

[382] Y. Liu, X. Lin, and S. Zhu. “Combined Coverage Path Planning for Autonomous Cleaning Robots in Unstructured Environments.” In: *Proceedings of the IEEE World Congress on Intelligent Control and Automation (WCICA)*. 2008, pp. 8271–8276 (↗ pp. 15, 32).

[383] K. J. Lohmann, N. F. Putman, and C. M. F. Lohmann. “The Magnetic Map of Hatchling Loggerhead Sea Turtles.” In: *Current Opinion in Neurobiology* 22.2 (2012), pp. 336–342 (↗ pp. 47, 61).

[384] S. G. Loizou and V. Kumar. “Biologically Inspired Bearing-Only Navigation and Tracking.” In: *Proceedings of the IEEE Conference on Decision and Control (CDC 07)*. Dec. 2007, pp. 1386–1391 (↗ p. 58).

[385] G. López-Nicolás, J. J. Gurrero, and C. Sagüés. “Multiple Homographies with Omnidirectional Vision for Robot Homing.” In: *Robotics and Autonomous Systems* 58.6 (201), pp. 773–783 (↗ p. 57).

[386] C. Losada, M. Mazo, S. E. Palazuelos, D. Pizarro, M. Marrón, and J. F. Velasco. “Identification and Tracking of Robots in an Intelligent Space Using Static Cameras and an XPFCP.” In: *Robotics and Autonomous Systems* 61.2 (2013), pp. 75–85 (↗ p. 107).

[387] C. Losada, M. Mazo, S. Palazuelos, D. Pizarro, and M. Marrón. “Multi-Camera Sensor System for 3D Segmentation and Localization of Multiple Mobile Robots.” In: *Sensors* 10.4 (2010), pp. 3261–3279 (↗ p. 107).

[388] D. G. Lowe. “Object Recognition from Local Scale-Invariant Features.” In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV 99)*. 1999, pp. 1150–1157 (↗ pp. 35, 41).

[389] D. G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints." In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110 (↗ pp. 15, 35, 41).

[390] B. D. Lucas and T. Kanade. "An Iterative Image Registration Technique with an Application to Stereo Vision." In: *Proceedings of the DARPA Image Understanding Workshop*. 1981, pp. 121–130 (↗ pp. 55, 56).

[391] C. Luo and S. J. Yang. "A Bioinspired Neural Network for Real-Time Concurrent Map Building and Complete Coverage Robot Navigation in Unknown Environments." In: *IEEE Transactions on Neural Networks* 19.7 (2008), pp. 1279–1298 (↗ p. 15).

[392] D. K. MacKinnon, V. Aitken, and F. Blais. "A Comparison of Precision and Accuracy in Triangulation Laser Range Scanners." In: *Proceedings of the Canadian Conference on Electrical and Computer Engineering*. 2006, pp. 832–837 (↗ p. 223).

[393] W. Maddern, M. Milford, and G. Wyeth. "Continuous Appearance-Based Trajectory SLAM." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 11)*. 2011, pp. 3595–3600 (↗ p. 77).

[394] W. Maddern, M. Milford, and G. Wyeth. "Capping Computation Time and Storage Requirements for Appearance-Based Localization with CAT-SLAM." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 12)*. 2012, pp. 822–827 (↗ pp. 35, 77).

[395] W. Maddern, M. Milford, and G. Wyeth. "CAT-SLAM: Probabilistic Localisation and Mapping Using a Continuous Appearance-Based Trajectory." In: *International Journal of Robotics Research* 31.4 (2012), pp. 429–451 (↗ pp. 35, 77).

[396] W. Maddern, M. Milford, and G. Wyeth. "Towards Persistent Localization and Mapping with a Continuous Appearance-Based Topology." In: *Proceedings of Robotics: Science and Systems (RSS 12)*. 2012, P36 (↗ pp. 35, 77).

[397] R. Madhavan, R. Lakaemper, and T. Kalmár-Nagy. "Benchmarking and Standardization of Intelligent Robotic Systems." In: *Proceedings of the IEEE International Conference on Advanced Robotics (ICAR 09)*. 2009, pp. 1–7 (↗ p. 88).

[398] D. Mahajan, R. Ramamoorthi, and B. Curless. "A Theory of Spherical Harmonic Identities for BRDF/Lighting Transfer and Image Consistency." In: *Proceedings of the European Conference on Computer Vision (ECCV 06)*. Ed. by A. Leonardis, H. Bischof, and A. Prinz. Vol. 3954. Lecture Notes in Computer Science (LNCS). Springer, 2006, pp. 41–55 (↗ pp. 45, 53).

[399] A. Makadia and K. Daniilidis. "Direct 3D-Rotation Estimation from Spherical Images via a Generalized Shift Theorem." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 03)*. June 2003, pp. 217–224 (↗ p. 45).

[400] A. Makadia and K. Daniilidis. "Rotation Recovery from Spherical Images Without Correspondences." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.7 (2006), pp. 1170–1175 (↗ p. 45).

[401] A. Makadia, L. Sorgi, and K. Daniilidis. "Rotation Estimation From Spherical Images." In: *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR 04)*. Aug. 2004, pp. 590–593 (↗ p. 45).

[402] A. Makadia, C. Geyer, and K. Daniilidis. "Correspondence-Free Structure from Motion." In: *International Journal of Computer Vision* 75.3 (Dec. 2007), pp. 311–327 (↗ p. 53).

[403] H. A. Mallot and K. Basten. "Embodied Spatial Cognition: Biological and Artificial Systems." In: *Image and Vision Computing* 27 (2009), pp. 1658–1670 (↗ pp. 3, 42, 61, 83, 86).

[404] M. Mangan and B. Webb. "Modelling Place Memory in Crickets." In: *Biological Cybernetics* 101.4 (Oct. 2009), pp. 307–323 (↗ pp. 52, 54).

[405] M. Manigandan and J. I. Manju. "Wireless Vision Based Mobile Robot Control Using Hand Gesture Recognition Through Perceptual Color Space." In: *Proceedings of the IEEE International Conference on Advances in Computer Engineering (ACE 10)*. 2010, pp. 95–99 (↗ pp. 29, 224).

[406] V. Mante, R. A. Frazor, V. Bonin, W. S. Geisler, and M. Carandini. "Independence of Luminance and Contrast in Natural Scenes and in the Early Visual System." In: *Nature Neuroscience* 8 (2005), pp. 1690–1697 (↗ p. 122).

[407] L. Maohai, S. Lining, H. Qingcheng, C. Zesu, and P. Songhao. "Robust Omnidirectional Vision Based on Mobile Robot Hierarchical Localization and Autonomous Navigation." In: *Information Technology Journal* 10.1 (2011), pp. 29–39 (↗ pp. 67, 69).

[408] G. L. Mariottini, E. Alunno, J. Piazzi, and D. Prattichizzo. "Epipole-Based Visual Servoing with Central Catadioptric Camera." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 05)*. Apr. 2005, pp. 3516–3521 (↗ pp. 46, 49, 57).

[409] G. L. Mariottini, D. Prattichizzo, and G. Oriolo. "Image-Based Visual Servoing for Nonholonomic Mobile Robots with Central Catadioptric Camera." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 06)*. May 2006, pp. 538–544 (↗ pp. 46, 49, 57).

[410] G. L. Mariottini, S. Scheggi, F. Morbidi, and D. Prattichizzo. "A Robust Uncalibrated Visual Compass Algorithm from Paracatadioptric Line Images." In: *Proceedings of the International Conference on Simulation, Modelling and Programming of Autonomous Robots (SIMPAR 08)*. 2008, pp. 242–255 (↗ p. 46).

[411] G. L. Mariottini and D. Prattichizzo. “Image-Based Visual Servoing with Central Catadioptric Cameras.” In: *International Journal of Robotics Research* 27.1 (Jan. 2008), pp. 41–56 (↗ pp. 46, 49, 57).

[412] G. L. Mariottini, S. Scheggi, F. Morbidi, and D. Prattichizzo. “An Accurate and Robust Visual-Compass Algorithm for Robot-Mounted Omnidirectional Cameras.” In: *Robotics and Autonomous Systems* 60.9 (Sept. 2012), pp. 1179–1190 (↗ p. 46).

[413] T. M. Martinez, S. G. Berkovich, and K. J. Schulten. “Neural Gas Network for Vector Quantization and its Applications to Time-Series Prediction.” In: *IEEE Transactions on Neural Networks* 4.4 (1993), pp. 558–569 (↗ p. 212).

[414] H. Martínez-Barberá and D. Herrero-Pérez. “Autonomous Navigation of an Automated Guided Vehicle in Industrial Environments.” In: *Robotics and Computer-Integrated Manufacturing* 26.4 (2010), pp. 296–311 (↗ p. 10).

[415] D. Marzorati, M. Matteucci, D. Migliore, and D. G. Sorrenti. “On the Collection of Robot-Pose Ground-Truth for Indoor Scenarios in the RAWSEEDS Project.” In: *In Proceedings of the IROS 08 Workshop on Benchmarks in Robotics Research.* 2008 (↗ p. 88).

[416] M. J. Matarić and F. Michaud. “Behavior-Based Systems.” In: *Springer Handbook of Robotics.* Springer, 2008, pp. 891–909 (↗ p. 220).

[417] J. Matas, O. Chum, M. Urban, and T. Pajdla. “Robust Wide-Baseline Stereo From Maximally Stable Extremal Regions.” In: *Proceedings of the British Machine Vision Conference (BMVC 02).* 2002, pp. 384–393 (↗ p. 35).

[418] C. McCarthy and N. Barnes. “A Unified Strategy for Landing and Docking Using Spherical Flow Divergence.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.5 (2012), pp. 1024–1031 (↗ pp. 29, 223).

[419] C. McCarthy, N. Barnes, and R. Mahony. “A Robust Docking Strategy for a Mobile Robot Using Flow Field Divergence.” In: *IEEE Transactions on Robotics* 24.4 (2008), pp. 832–842 (↗ p. 29).

[420] J. Mehlhorn and G. Rehkämper. “Neurobiology of the Homing Pigeon. A Review.” In: *Naturwissenschaften* 96.9 (2009), pp. 1011–1025 (↗ pp. 47, 61).

[421] E. Menegatti, T. Maeda, and H. Ishiguro. “Image-Based Memory for Robot Navigation Using Properties of Omnidirectional Images.” In: *Robotics and Autonomous Systems* 47.4 (2004), pp. 251–267 (↗ pp. 32, 39, 77, 78, 81, 82, 129, 170, 173, 174).

[422] E. Menegatti, A. Pretto, and E. Pagello. “Testing Omnidirectional Vision-Based Monte Carlo Localization Under Occlusion.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 04).* 2004, pp. 2487–2493 (↗ pp. 75, 89).

[423] E. Menegatti, A. Pretto, A. Scarpa, and E. Pagello. “Omnidirectional Vision Scan Matching for Robot Localization in Dynamic Environments.” In: *IEEE Transactions on Robotics* 22.3 (2006), pp. 523–535 (↗ pp. 26, 39, 75, 89).

[424] E. Menegatti, G. Aneloni, M. Wright, and E. Pagello. “The Spatial Semantic Hierarchy Implemented with an Omnidirectional Vision System.” In: *Cybernetics and Systems* 39.4 (2008), pp. 443–466 (↗ pp. 68, 79).

[425] E. Menegatti, M. Wright, and E. Pagello. “A New Omnidirectional Vision Sensor for the Spatial Semantic Hierarchy.” In: *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics.* 2001, pp. 93–98 (↗ p. 26).

[426] E. Menegatti, M. Zoccarato, E. Pagello, and H. Ishiguro. “Hierarchical Image-Based Localisation for Mobile Robots with Monte-Carlo Localisation.” In: *Proceedings of the European Conference on Mobile Robots (ECMR 03).* 2003, pp. 13–20 (↗ pp. 3, 32, 39, 78).

[427] E. Menegatti, M. Zoccarato, E. Pagello, and H. Ishiguro. “Image-Based Monte Carlo Localisation with Omnidirectional Images.” In: *Robotics and Autonomous Systems* 48.1 (2004), pp. 17–30 (↗ pp. 32, 39, 78, 170).

[428] R. Menzel and M. Giurfa. “Cognitive Architecture of a Mini-Brain: The Honeybee.” In: *Trends in Cognitive Sciences* 5.2 (2001), pp. 62–71 (↗ p. 86).

[429] R. Menzel, U. Greggers, A. Smith, S. Berger, R. Brandt, S. Brunke, G. Bundrock, S. Hülse, T. Plümpe, F. Schaupp, E. Schüttler, S. Stach, J. Stindt, N. Stollhoff, and S. Watzl. “Honey Bees Navigate According to a Map-Like Spatial Memory.” In: *Proceedings of the National Academy of Sciences of the United States of America* 102.8 (2005), pp. 3040–3045 (↗ p. 86).

[430] R. Menzel, A. Kirbach, W.-D. Haass, B. Fischer, J. Fuchs, M. Koblofsky, K. Lehmann, L. Reiter, H. Meyer, H. Nguyen, S. Jones, P. Norton, and U. Greggers. “A Common Frame of Reference for Learned and Communicated Vectors in Honeybee Navigation.” In: *Current Biology* 21.8 (2011), pp. 645–650 (↗ p. 86).

[431] R. Menzel, K. Lehmann, G. Manz, J. Fuchs, M. Koblofsky, and U. Greggers. “Vector Integration and Novel Shortcutting in Honeybee Navigation.” In: *Apidologie* 43.3 (2012), pp. 229–243 (↗ p. 86).

[432] J.-A. Meyer and D. Filliat. “Map-Based Navigation in Mobile Robots. Part II: A Review of Map-Learning and Path-Planning Strategies.” In: *Cognitive Systems Research* 4.4 (2003), pp. 283–317 (↗ pp. 5, 17, 20–22, 40, 62, 64, 66, 76, 80, 85).

[433] J.-A. Meyer and A. Guillot. “Biologically Inspired Robots.” In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 1395–1422 (↗ pp. 60, 66).

[434] Y. Mezouar, H. Hadj Abdelkader, P. Martinet, and F. Chaumette. “Central Catadioptric Visual Servoing from 3D Straight Lines.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2004, pp. 343–348 (↗ pp. 35, 49, 57).

[435] D. Michel, A. A. Argyros, and M. L. A. Lourakis. “Localizing Unordered Panoramic Images Using the Levenshtein Distance.” In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV 07)*. 2007, pp. 1–7 (↗ pp. 39, 74).

[436] D. Michel, A. A. Argyros, and M. I. A. Lourakis. “Horizon Matching for Localizing Unordered Panoramic Images.” In: *Computer Vision and Image Understanding* 114.2 (2010), pp. 274–285 (↗ pp. 39, 74).

[437] B. Mičušík and J. Košecká. “Piecewise Planar City 3D Modeling from Street View Panoramic Sequences.” In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 09)*. 2009, pp. 2906–2912 (↗ p. 73).

[438] K. Mikolajczyk and C. Schmid. “A Performance Evaluation of Local Descriptors.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.10 (2005), pp. 1615–1630 (↗ pp. 33, 130, 169).

[439] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. “A Comparison of Affine Region Detectors.” In: *International Journal of Computer Vision* 65.1-2 (2005), pp. 43–72 (↗ pp. 33, 169).

[440] M. Milford. “Visual Route Recognition with a Handful of Bits.” In: *Proceedings of Robotics: Science and Systems (RSS 12)*. 2012, P38 (↗ pp. 21, 82).

[441] M. Milford and G. Wyeth. “Persistent Navigation and Mapping Using a Biologically Inspired SLAM System.” In: *International Journal of Robotics Research* 29.9 (2010), pp. 1131–1153 (↗ pp. 30, 77, 78, 89).

[442] J. Minguez, F. Lamiraux, and J.-P. Laumond. “Motion Planning and Obstacle Avoidance.” In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 827–852 (↗ p. 66).

[443] J. Miura, Y. Negishi, and Y. Shirai. “Mobile Robot Map Generation by Integrating Omnidirectional Stereo and Laser Range Finder.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 02)*. 2002, pp. 250–255 (↗ pp. 40, 74, 75).

[444] J. Miura and S. Ikeda. “A Simple Modelling of Complex Environments for Mobile Robots.” In: *International Journal of Intelligent Systems Technologies and Applications* 6.1 (2009), pp. 166–177 (↗ pp. 73, 74).

[445] T. Miyake and H. Ishihara. “Mechanisms and Basic Properties of Window Cleaning Robot.” In: *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 03)*. Vol. 2. 2003, pp. 1372–1377 (↗ p. 10).

[446] R. Möller. “Insect Visual Homing Strategies in a Robot With Analog Processing.” In: *Biological Cybernetics* 83.3 (2000), pp. 231–243 (↗ pp. 53, 102).

[447] R. Möller. “Local Visual Homing by Warping of Two-Dimensional Images.” In: *Robotics and Autonomous Systems* 57.1 (2009), pp. 87–101 (↗ pp. 30, 50, 51, 58, 97, 103, 142).

[448] R. Möller. *A Comparison of Four Distance Measures for 2d Warping Methods.* Technical report. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2010 (↗ pp. 22, 126, 235, 236, 239).

[449] R. Möller. *An Alternative Distance Measure for 2d Warping Methods.* Technical report. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2010 (↗ pp. 22, 125, 126, 235, 239).

[450] R. Möller. *Illumination Tolerance for a Holistic Visual Navigation Method: Tunable Distance Measures and Two Novel Forms of Correlation.* Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2013 (↗ pp. 22, 166, 167, 218).

[451] R. Möller, M. Krzykawski, and L. Gerstmayr. “Three 2d-Warping Schemes for Visual Robot Navigation.” In: *Autonomous Robots* 29.3 (2010), pp. 253–291 (↗ pp. 30, 47, 50–52, 56, 58, 61, 97, 98, 103, 105, 114, 142, 219).

[452] R. Möller and A. Vardy. “Local Visual Homing by Matched-Filter Descent in Image Distances.” In: *Biological Cybernetics* 95.5 (2006), pp. 413–430 (↗ pp. 50, 52, 56, 114, 142).

[453] R. Möller, A. Vardy, L. Gerstmayr, F. Röben, and S. Kreft. “Neuroethological Concepts at Work: Insect-Inspired Methods for Visual Robot Navigation.” In: *Biological Approaches for Engineering*. 2008, pp. 91–94 (↗ pp. 30, 61, 89, 91, 213, 216).

[454] R. Möller, A. Vardy, S. Kreft, and S. Ruwisch. “Visual Homing in Environments with Anisotropic Landmark Distribution.” In: *Autonomous Robots* 23.3 (2007), pp. 231–245 (↗ pp. 52, 56, 88, 142).

[455] R. Möller. “Insects Could Exploit UV-Green Contrast for Landmark Navigation.” In: *Journal of Theoretical Biology* 214.4 (2002), pp. 619–631 (↗ pp. 60, 217).

[456] R. Möller. “A Model of Ant Navigation Based on Visual Prediction.” In: *Journal of Theoretical Biology* 305 (2012), pp. 118–130 (↗ pp. 30, 51, 56, 216).

[457] R. Möller, M. Krzykawski, L. Gerstmayr-Hillen, M. Horst, D. Fleer, and J. de Jong. "Cleaning Robot Navigation Using Panoramic Views and Particle Clouds as Landmarks." In: *Robotics and Autonomous Systems* 61.12 (2013), pp. 1415–1439 (↗ pp. 9, 39, 79, 88, 89, 108, 113, 214, 221, 222).

[458] M. Montemerlo and S. Thrun. *FastSLAM: A Scalable Method for the Simultaneous Localization and Mapping Problem in Robotics.* Springer Tracts in Advanced Robotics (STAR). Springer, 2007 (↗ p. 39).

[459] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. "FastSLAM: a Factored Solution to the Simultaneous Localization and Mapping Problem." In: *Eighteenth National Conference on Artificial Intelligence.* 2002, pp. 593–598 (↗ p. 39).

[460] M. Montemerlo, S. Thrun, D. Roller, and B. Wegbreit. "FastSLAM 2.0: An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping That Provably Converges." In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 03).* 2003, pp. 1151–1156 (↗ p. 39).

[461] H. Moravec and A. E. Elfes. "High Resolution Maps from Wide Angle Sonar." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 85).* 1985, pp. 116–121 (↗ p. 74).

[462] P. Morin and C. Samson. "Motion Control of Wheeled Mobile Robots." In: *Springer Handbook of Robotics.* Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 799–826 (↗ pp. 66, 70, 72, 75, 78, 82, 113).

[463] R. G. M. Morris. "Spatial Localization Does Not Require the Presence of Local Cues." In: *Learning and Motivation* 12 (1981), pp. 239–260 (↗ p. 61).

[464] R. G. M. Morris. "Developments of a Water-Maze Procedure for Studying Spatial Learning in the Rat." In: *Journal of Neuroscience Methods* 11 (1984), pp. 47–60 (↗ p. 61).

[465] E. I. Moser, E. Kropff, and M.-B. Moser. "Place Cells, Grid Cells, and the Brain's Spatial Representation System." In: *Annual Reviews of Neuroscience* 31 (2008), pp. 69–89 (↗ pp. 66, 86).

[466] H. Mouritsen. "Sensory Biology: Search for the Compass Needles." In: *Nature* 484.7394 (2012), p. 320 (↗ pp. 47, 61).

[467] R. Muheim. "Behavioural and Physiological Mechanisms of Polarized Light Sensitivity in Birds." In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 366.1565 (2011), pp. 763–771 (↗ pp. 47, 61).

[468] R. Muheim, J. B. Phillips, and M. E. Deutschlander. "White-Throated Sparrows Calibrate Their Magnetic Compass by Polarized Light Cues During Both Autumn and Spring Migration." In: *Journal of Experimental Biology* 212.21 (2009), pp. 3466–3472 (↗ pp. 47, 61).

[469] M. Muja and D. G. Lowe. "Fast Matching of Binary Features." In: *Proceedings of the IEEE Conference on Computer and Robot Vision (CRV 12).* 2012, pp. 404–410 (↗ p. 41).

[470] C. Munier. "Anpassung von Strategien zur flächendeckenden Exploration von Räumen an einen Reinigungsroboter." Bachelor's Thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2008. In German. (↗ pp. 17, 220).

[471] H. Murase and S. K. Nayar. "Visual Learning and Recognition of 3D Objects from Appearance." In: *International Journal on Computer Vision* 14.1 (1995), pp. 5–24 (↗ pp. 20, 59).

[472] A. C. Murillo, P. Abad, J. J. Guerrero, and C. Sagüés. "Improving Topological Maps for Safer and Robust Navigation." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 09).* 2009, pp. 3609–3614 (↗ pp. 26, 77).

[473] A. C. Murillo, P. Campos, J. Košecká, and J. J. Guerrero. "Gist Vocabularies in Omnidirectional Images for Appearance Based Mapping and Localization." In: *Proceedings of the Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras (OMNIVIS 10).* 2010 (↗ pp. 32, 33, 36, 69, 81).

[474] A. C. Murillo, J. J. Guerrero, and C. Sagüés. "SURF Features for Efficient Robot Localization with Omnidirectional Images." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 07).* 2007, pp. 3901–3907 (↗ pp. 68, 78).

[475] A. C. Murillo and J. Košecká. "Experiments in Place Recognition Using GIST Panoramas." In: *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW 09).* 2009, pp. 2196–2203 (↗ pp. 32, 33, 35, 36, 68, 69, 211).

[476] A. C. Murillo, C. Sagüés, J. J. Guerrero, and Goedemé. "From Omnidirectional Images to Hierarchical Localization." In: *Robotics and Autonomous Systems* 55.5 (2007), pp. 372–382 (↗ pp. 26, 32, 35, 36, 67, 68, 72, 78, 170, 172).

[477] A. C. Murillo, C. Sagüés, J. J. Guerrero, T. Goedemé, T. Tuytelaars, and L. Van Gool. "Hierarchical Localization by Matching Vertical Lines in Omnidirectional Images." In: *Proceedings of the IEEE/RSJ International Workshop — from Sensors to Human Spatial Concepts (IROS 06).* 2006 (↗ pp. 26, 32, 35, 36, 68, 72, 78).

[478] K. Murphy and S. Russell. "Rao-Blackwellized Particle Filtering for Dynamic Bayesian Networks." In: *Sequential Monte-Carlo Methods in Practise.* Ed. by A. Doucet, N. de Freitas, and N. Gordon. Statistics for Engineering and Information Science. Springer, 2001, pp. 499–515 (↗ p. 39).

[479] A. Nabatchian, E. Abdel-Raheem, and M. Ahmadi. "Illumination Invariant Feature Extraction and Mutual-Information-Based Local Matching for Face Recognition Under Illumination Variation and Occlusion." In: *Pattern Recognition* 44.10-11 (2011), pp. 2576–2587 (↗ p. 128).

[480] D. A. Najera, E. L. McCullough, and R. Jander. “Interpatch Foraging in Honeybees-Rational Decision Making at Secondary Hubs Based Upon Time and Motivation.” In: *Animal Cognition* 15.6 (2012), pp. 1195–1203 (↗ p. 86).

[481] M. Neal and F. Labrosse. “Rotation-Invariant Appearance Based Maps for Robot Navigation Using an Artificial Immune Network Algorithm.” In: *Proceedings of the Congress on Evolutionary Computation (CEC 04).* 2004, pp. 863–870 (↗ p. 81).

[482] S. Negahdaripour. “Revised Definition of Optical Flow: Integration of Radiometric and Geometric Cues for Dynamic Scene Analysis.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20.9 (1998), pp. 961–979 (↗ p. 130).

[483] Y. Negishi, J. Miura, and Y. Shirai. “Vision-Based Mobile Robot Speed Control Using a Probabilistic Occupancy Map.” In: *Proceedings of the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 03).* 2003, pp. 64–69 (↗ pp. 74, 75).

[484] Y. Negishi, J. Miura, and Y. Shirai. “Calibration of Omnidirectional Stereo for Mobile Robots.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 04).* 2004, pp. 2600–2605 (↗ p. 26).

[485] Y. Negishi, J. Miura, and Y. Shirai. “Mobile Robot Navigation in Unknown Environments Using Omnidirectional Stereo and Laser Range Finder.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 04).* Vol. 3. 2004, pp. 2737–2742 (↗ pp. 40, 74, 75).

[486] R. Nelson and J. Aloimonos. “Finding Motion Parameters from Spherical Motion Fields (or the Advantage of Having Eyes in the Back of Your Head).” In: *Biological Cybernetics* 58.4 (1988), pp. 261–273 (↗ pp. 28, 45).

[487] H. Neumann, L. Pessoa, and T. Hansen. “Interaction of ON and OFF Pathways for Visual Contrast Measurement.” In: *Biological Cybernetics* 81.5-6 (1999), pp. 515–532 (↗ p. 122).

[488] P. Newman, G. Sibley, M. Smith, M. Cummins, A. Harrison, C. Mei, I. Posner, R. Shade, D. Schroeter, L. Murphy, W. Churchill, D. Cole, and I. Reid. “Navigating, Recognizing and Describing Urban Spaces with Vision and Lasers.” In: *International Journal of Robotics Research* 28.11-12 (2009), pp. 1406–1433 (↗ pp. 35, 81, 88).

[489] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. H. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. “The QBIC Project: Querying Images by Content, Using Color, Texture, and Shape.” In: *Storage and Retrieval for Image and Video Databases.* Ed. by W. Niblack. Proceedings of the SPIE. 1993, pp. 173–187 (↗ p. 177).

[490] D. Nister. “An Efficient Solution to the Five-Point Relative Pose Problem.” In: *Proceedings of the International IEEE Conference on Computer Vision and Pattern Recognition (CVPR 03).* 2003, pp. 560–567 (↗ p. 57).

[491] S. Y. Nof, ed. *Springer Handbook of Automation.* Springer, 2009 (↗ p. 1).

[492] A. Oliva. “Gist of the Scene.” In: *The Encyclopedia of Neurobiology of Attention.* Ed. by L. Itti, G. Rees, and J. K. Tsotsos. Elsevier, 2005, pp. 251–256 (↗ p. 33).

[493] A. Oliva and A. Torralba. “Building the Gist of a Scene: The Role of Global Image Features in Recognition.” In: *Progress in Brain Research* 155.B (2006), pp. 23–36 (↗ p. 33).

[494] B. A. Olshausen and D. J. Field. “Natural Image Statistics and Efficient Coding.” In: *Network: Computation in Neural Systems* 7.2 (1996), pp. 333–339 (↗ p. 122).

[495] B. Olshausen. “Principles of Image Representation in Visual Cortex.” In: *The Visual Neurosciences.* MIT Press, 2003, pp. 1603–1615 (↗ pp. 122, 129, 174).

[496] R. Orghidan, E. M. Mouaddib, J. Salvi, and J. J. Serrano. “Catadioptric Single-Shot Rangefinder for Textured Map Building in Robot Navigation.” In: *IET Computer Vision* 1.2 (2007), pp. 43–53 (↗ p. 223).

[497] M. Pahl, H. Zhu, J. Tautz, and S. Zhang. “Large Scale Homing in Honeybees.” In: *PloS One* 6.5 (2011), e19669 (↗ p. 86).

[498] T. Pajdla, T. Svoboda, and V. Hlaváč. “Epipolar Geometry of Central Panoramic Catadioptric Cameras.” In: *Panoramic Vision: Sensors, Theory, and Applications.* Ed. by R. Benosman and S. B. Kang. Springer, 2001, pp. 73–102 (↗ pp. 24, 26).

[499] J. Palacín, J. A. Salse, I. Valgañón, and X. Clua. “Building a Mobile Robot for a Floor-Cleaning Operation in Domestic Environments.” In: 53.5 (2004), pp. 1418–1424 (↗ p. 14).

[500] L. Paletta, S. Frintrop, and J. Hertzberg. “Robust Localization Using Context in Omnidirectional Imaging.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 01).* Vol. 2. 2001, pp. 2072–2077 (↗ pp. 32, 33, 82).

[501] T. Palleja, M. Tresanchez, M. Teixido, and J. Palacín. “Modeling Floor-Cleaning Coverage Performances of Some Domestic Mobile Robots in a Reduced Scenario.” In: *Robotics and Autonomous Systems* 58.1 (2010), pp. 37–45 (↗ pp. 13, 15, 108).

[502] B. Pan, H. Xie, and Z. Wang. “Equivalence of Digital Image Correlation Criteria for Pattern Matching.” In: *Applied Optics* 49.2 (2010), pp. 5501–5509 (↗ p. 127).

[503] G. Panin and A. Knoll. “Mutual Information-Based 3D Object Tracking.” In: *International Journal of Computer Vision* 78.1 (2008), pp. 107–118 (↗ p. 128).

[504] T. V. Papathomas, C. Chubb, A. Gorea, and E. Kowler. *Early Vision and Beyond.* MIT Press, 1995 (↗ p. 122).

[505] L. Payá and L. Fernández. "Appearance-Based Dense Maps Creation. Comparison of Compression Techniques with Panoramic Images." In: *Proceedings of the International Conference on Informatics in Control, Automation, and Robotics (ICINCO 09).* 2009, pp. 250–255 (↗ pp. 3, 32, 77, 78).

[506] L. Payá, L. Fernández, A. Gil, and Ó. Reinoso. "Map Building and Monte-Carlo Localization Using Global Appearance of Omnidirectional Images." In: *Sensors* 10.12 (2010), pp. 11468–11497 (↗ pp. 3, 32, 39, 77, 78, 89, 170).

[507] L. M. Paz, J. D. Tardós, and J. Neira. "Divide and Conquer: EKF SLAM in O(N)." In: *IEEE Transactions on Robotics* 24.5 (2008) (↗ pp. 40, 71).

[508] O. Pele and M. Werman. "Fast and Robust Earth Mover's Distances." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV 09).* 2009, pp. 460–467 (↗ pp. 33, 178, 199, 209).

[509] O. Pele and M. Werman. "The Quadratic-Chi Histogram Distance Family." In: *Proceedings of the European Conference on Computer Vision (ECCV 10).* Ed. by K. Daniilidis, P. Maragos, and N. Paragios. Vol. 6312. Lecture Notes in Computer Science (LNCS). Sprinter, 2010, pp. 749–762 (↗ pp. 178, 180, 199, 209).

[510] H. Peng, F. Long, and C. Ding. "Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.8 (2005), pp. 1226–1238 (↗ p. 128).

[511] A. Philippides, B. Baddeley, K. Cheng, and P. Graham. "How Might Ants Use Panoramic Views for Route Navigation?" In: *Journal of Experimental Biology* 214 (2011), pp. 445–451 (↗ p. 86).

[512] A. Philippides, B. Baddeley, P. Husbands, and P. Graham. "How Can Embodiment Simplify the Problem of View-Based Navigation?" In: *Living Machines.* Ed. by T. J. Prescott, N. F. Lepora, A. Mura, and P. F. M. J. Verschure. Vol. 7375. Lecture Notes in Computer Science (LNCS). Springer, 2012, pp. 216–227 (↗ pp. 83, 84, 86).

[513] C. Pirchheim and G. Reitmayr. "Homography-Based Planar Mapping and Tracking for Mobile Phones." In: *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality (ISMAR 11).* 2011, pp. 27–36 (↗ p. 85).

[514] C. Plagemann, F. Endres, J. Hess, C. Stachniss, and W. Burgard. "Monocular Range Sensing: A Non-Parametric Learning Approach." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 08).* 2008, pp. 929–934 (↗ pp. 74, 75).

[515] C. Plagemann, C. Stachniss, J. Hess, F. Endres, and N. Franklin. "A Nonparametric Learning Approach to Range Sensing from Omnidirectional Vision." In: *Robotics and Autonomous Systems* 58.6 (2010), pp. 762–772 (↗ pp. 74, 75).

[516] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever. "Mutual Information Based Registration of Medical Images: A Survey." In: *IEEE Transactions on Medical Imaging,* 22.8 (2003), pp. 986–1004 (↗ pp. 128, 155).

[517] I. Powell. "Panoramic Lens." In: *Applied Optics* 33.31 (1994), pp. 7356–7361 (↗ p. 26).

[518] E. Prassler and K. Kosuge. "Domestic Robotics." In: *Springer Handbook of Robotics.* Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 1253–1281 (↗ pp. 1, 3, 10–13).

[519] E. Prassler, A. Ritter, C. Schäffer, and P. Fiorini. "A Short History of Cleaning Robots." In: *Autonomous Robots* 9.3 (2000), pp. 211–226 (↗ pp. 10, 11, 14).

[520] E. Prassler. "Servicerobotik und die Entdeckung ihrer Langsamkeit." In: *Economic Engineering* 5 (2010), pp. 33–36. In German. (↗ p. 1).

[521] E. Prassler, M. Hägele, and R. Siegwart. "International Contest for Cleaning Robots: Fun Event or a First Step Towards Benchmarking Service Robots." In: *Field and Service Robotics. Recent Advances in Reserch and Applications.* Ed. by S. Yuta, H. Asama, S. Thrun, E. Prassler, and T. Tsubouchi. Vol. 24. Springer Tracts in Advanced Robotics (STAR). Springer, 2006, pp. 447–456 (↗ p. 14).

[522] A. Pronobis and B. Caputo. "COLD. COsy Localization Database." In: *International Journal of Robotics Research* 28.5 (2009), pp. 588–594 (↗ p. 88).

[523] F. Provost and T. Fawcett. "Robust Classification for Imprecise Environments." In: *Machine Learning* 42.3 (2001), pp. 203–231 (↗ pp. 138, 139, 182).

[524] J. Puzicha, T. Hoffmann, and J. Buhmann. "Non-Parametric Similarity Measures for Unsupervised Texture Segmentation and Image Retrieval." In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (ICPR 97).* 1997, pp. 267–272 (↗ p. 176).

[525] Z. Qian, Y. Zhao, and Z. Fu. "Development of Wall-Climbing Robots with Sliding Suction Cups." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 06).* 2006, pp. 3417–3422 (↗ p. 10).

[526] B. Qiu, G. Qian, Z. Xiang, and Z. Li. "Avoiding Barriers in Control of Mowing Robot." In: *Frontiers of Mechanical Engineering in China* 1.3 (2006), pp. 346–349 (↗ p. 10).

[527] N. Ragot, R. Rossi, X. Savatier, J. Ertaud, and B. Mazari. "3D Volumetric Reconstruction with a Catadioptric Stereovision Sensor." In: *Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE 08).* 2008, pp. 1306–1311 (↗ pp. 74, 75).

[528] A. Ramisa, A. Goldhoorn, D. Aldavert, R. Toledo, and R. L. de Mantaras. "Combining Invariant Features and the ALV Homing Method for Autonomous Robot Navigation Based on Panoramas." In: *Journal of Intelligent and Robotic Systems* 64.3–4 (2011), pp. 625–649 (↗ p. 54).

[529] A. Ramisa, A. Tapus, D. Aldavert, R. Toledo, and R. Lopez de Mantaras. "Robust Vision-Based Robot Localization Using Combinations of Local Feature Region Detectors." In: *Autonomous Robots* 27.4 (2009), pp. 373–385 (↗ pp. 35, 36, 82).

[530] A. Ranganathan, E. Menegatti, and F. Dellaert. "Bayesian Inference in the Space of Topological Maps." In: *IEEE Transactions on Robotics* 22.1 (2006), pp. 92–107 (↗ pp. 32, 39, 81, 170).

[531] A. Ranganathan and F. Dellaert. "Online Probabilistic Topological Mapping." In: *International Journal of Robotics Research* 30.6 (2011), pp. 755–771 (↗ pp. 35, 81).

[532] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning.* MIT Press, 2006 (↗ p. 16).

[533] A. C. Rencher. *Methods of Multivariate Statistics.* 2nd ed. Wiley Interscience, 2002 (↗ pp. 33, 35, 212).

[534] S. M. Reppert, R. J. Gegear, and C. Merlin. "Navigational Mechanisms of Migrating Monarch Butterflies." In: *Trends in Neurosciences* 33.9 (2010), pp. 399–406 (↗ pp. 47, 61).

[535] S. Rhim, J. Ryu, K. Park, and S. Lee. "Performance Evaluation Criteria for Autonomous Cleaning Robots." In: *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA 07).* 2007, pp. 167–172 (↗ pp. 13, 108).

[536] O. Riabinina, N. H. d. Ibarra, L. Howard, and T. S. Collett. "Do Wood Ants Learn Sequences of Visual Stimuli?" In: *Journal of Experimental Biology* 214.16 (2011), pp. 2739–2748 (↗ p. 86).

[537] F. Röben. "Biologically Inspired Visual Navigation in Indoor and Outdoor Environments." PhD thesis. Faculty of Technology, Bielefeld University, Germany, 2010 (↗ pp. 53, 60, 91, 172, 173, 210).

[538] L. F. Rojo, L. Payá, O. Reinoso, J. M. Marín, and A. Gil. "Comparison of Mapping Techniques in Appearance-Based Topological Maps Creation." In: *Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation (ETFA 10).* 2010, pp. 1–7 (↗ pp. 3, 32, 39, 77, 78, 89).

[539] E. Rome, J. Hertzberg, F. Kirchner, U. Licht, and T. Christaller. "Towards Autonomous Sewer Robots: The MAKRO Project." In: *Urban Water* 1.1 (1999), pp. 57–70 (↗ p. 10).

[540] S. Ross, J. Pineau, B. Chaib-draa, and P. Kreitmann. "A Bayesian Approach for Learning and Planning in Partially Observable Markov Decision Processes." In: *Journal of Machine Learning Research* 12.3 (2011), pp. 1729–1770 (↗ p. 39).

[541] F. Rossi, A. Ranganathan, F. Dellaert, and E. Menegatti. "Toward Topological Localization with Spherical Fourier Transform and Uncalibrated Camera." In: *Proceedings of the International Conference on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR 08).* 2008, pp. 319–330 (↗ pp. 32, 82).

[542] E. Rosten and T. Drummond. "Machine Learning for High-Speed Corner Detection." In: *Proceedings of the European Conference on Computer Vision (ECCV 06).* Ed. by A. Leonardis, H. Bischof, and A. Prinz. Vol. 3951. Lecture Notes in Computer Science (LNCS). Springer, 2006, pp. 430–443 (↗ p. 41).

[543] E. Rosten, R. Porter, and T. Drummond. "Faster and Better: A Machine Learning Approach to Corner Detection." In: *IEEE Trans. Pattern Analysis and Machine Intelligence* 32.1 (2010), pp. 105–119 (↗ p. 41).

[544] P. M. Roth and M. Winter. *Survey of Appearance-Basd Methods for Object Recognition.* Technical report. Institute for Computer Graphics and Vision, Graz University of Technology, Austria, 2008 (↗ p. 20).

[545] A. Rottmann, O. M. Mozos, C. Stachniss, and W. Burgard. "Semantic Place Classification of Indoor Environments with Mobile Robots Using Boosting." In: *Proceedings of the National Conference on Artificial Intelligence.* Vol. 20. 3. 2005, p. 1306 (↗ pp. 211, 212).

[546] S. Roweis and L. Saul. "Nonlinear Dimensionality Reduction by Locally Linear Embedding." In: *Science* 290.5500 (2000), pp. 2323–2326 (↗ p. 212).

[547] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. "ORB: An Efficient Alternative to SIFT or SURF." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV 11).* 2011, pp. 2564–2571 (↗ p. 41).

[548] Y. Rubner, C. Tomasi, and L. J. Guibas. "A Metric for Distributions with Applications to Image Databases." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV 98).* 1998, pp. 59–66 (↗ p. 178).

[549] Y. Rubner, C. Tomasi, and L. J. Guibas. "The Earth Mover's Distance as a Metric for Image Retrieval." In: *International Journal of Computer Vision* 40.2 (2000), pp. 99–121 (↗ pp. 33, 119, 123, 172, 176–178, 180).

[550] S. Ruwisch. "Systematische Strategien auf topologischen Karten zur flächendeckenden Exploration von Räumen." Diploma Thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2007. In German. (↗ pp. 17, 220).

[551] P. E. Rybski, S. I. Roumeliotis, M. Gini, and N. Papanikolopoulos. "A Comparison of Maximum Likelihood Methods for Appearance-Based Minimalistic SLAM." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 04).* 2004, pp. 1777–1782 (↗ pp. 35, 39, 77).

[552] P. E. Rybski, S. Roumeliotis, M. L. Gini, and N. Papanikopoulos. “Appearance-Based Mapping Using Minimalistic Sensor Models.” In: *Autonomous Robots* 24.3 (2008), pp. 229–246 (↗ p. 77).

[553] P. E. Rybski, F. Zacharias, M. L. Gini, and N. Papanikolopoulos. “Using Visual Features for Building and Localizing Within Topological Maps of Indoor Environments.” In: *Innovations in Robot Mobility and Control.* Ed. by S. Patnaik, S. G. Tzafestas, G. Resconi, and A. Konar. Vol. 8. Studies in Computational Intelligence. Springer, 2005, pp. 251–271 (↗ pp. 35, 77).

[554] P. E. Rybski, F. Zacharias, and J.-F. Lett. “Using Visual Features to Build Topological Maps of Indoor Environments.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 03).* Vol. 1. 2003, pp. 850–855 (↗ pp. 35, 39, 77).

[555] D. G. Sabatta. “Vision-Based Topological Map Building and Localisation Using Persistent Features.” In: *Proceedings of the Robotics and Mechatronics Symposium (CSIR 08).* 2008, pp. 1–6 (↗ pp. 81, 82).

[556] H. Sadjadi and M. A. Jarrah. “Autonomous Cleaning System for Dubai International Airport.” In: *Journal of the Franklin Institute* 348.1 (2011), pp. 112–124 (↗ p. 10).

[557] M. Saedan, C. W. Lim, and M. H. Ang. “Omnidirectional Image Matching for Vision-Based Robot Localization.” In: *Proceedings of the IEEE International Conference on Mechatronics (ICM 06).* 2006, pp. 17–22 (↗ p. 77).

[558] M. Saedan, C. W. Lim, and M. H. Ang. “Appearance-Based SLAM with Map Loop Closing Using an Omnidirectional Camera.” In: *Proceedings of the IEEE/ASME International Conference Advanced Intelligent Mechatronics (ICAIM 07).* 2007, pp. 1–6 (↗ p. 77).

[559] M. Saedan, C. Lim, and M. Ang. “Vision-Based Localization Using a Central Catadioptric Vision System.” In: *Experimental Robotics.* Ed. by O. Khatib, V. Kumar, and D. Rus. Vol. 39. Springer Tracts in Advanced Robotics (STAR). Springer, 2008, pp. 397–406 (↗ p. 77).

[560] J. Saez Pons, W. Hübner, H. Dahmen, and H. A. Mallot. “Vision-Based Robot Homing in Dynamic Environments.” In: *Proceedings of the 13th IASTED International Conference on Robotics and Applications.* 2007, pp. 293–298 (↗ pp. 35, 46, 57, 58).

[561] C. Sagüés, A. C. Murillo, J. J. Guerrero, T. Goedemé, T. Tuytelaars, and L. Van Gool. “Localization with Omnidirectional Images Using the Radial Trifocal Tensor.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 06).* 2006, pp. 551–556 (↗ pp. 26, 35, 68, 72, 78).

[562] H. Sahin and L. Guvenc. “Household Robotics: Autonomous Devices for Vacuuming and Lawn Mowing.” In: *IEEE Control Systems Magazine* 27.2 (2007), pp. 20–96 (↗ pp. 9, 10).

[563] D. Scaramuzza and F. Fraundorfer. “Visual Odometry. Part I: the First 30 Years and Fundamentals.” In: *IEEE Robotics and Automation Magazine* 18.4 (2011), pp. 80–92 (↗ pp. 3, 17, 36, 57, 65, 72, 93, 113).

[564] D. Scaramuzza, A. Martinelli, and R. Siegwart. “A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion.” In: *Proceedings of the IEEE International Conference on Computer Vision Systems (ICVS 06).* 2006, p. 45 (↗ p. 24).

[565] D. Scaramuzza, A. Martinelli, and R. Siegwart. “A Toolbox for Easily Calibrating Omnidirectional Cameras.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 06).* 2006, pp. 5695–5701 (↗ p. 24).

[566] D. Scaramuzza. “Omnidirectional Camera.” In: *Encyclopedia of Computer Vision.* Ed. by K. Ikeuchi. Springer, 2013, pages not yet available. url: `http://robotics.ethz.ch/~scaramuzza/Davide_Scaramuzza_files/publications/pdf/omnidirectional_camera.pdf` (visited on 2013-10-17). To appear. (↗ pp. 23–25).

[567] D. Scaramuzza, N. Criblez, A. Martinelli, and R. Siegwart. “Robust Feature Extraction and Matching for Omnidirectional Images.” In: *Field and Service Robotics. Results of the 6th International Conference.* Ed. by C. Laugier and R. Siegwart. Vol. 42. Springer Tracts on Advanced Robotics (STAR). 2008, pp. 71–81 (↗ p. 35).

[568] D. Scaramuzza, F. Fraundorfer, and M. Pollefeys. “Closing the Loop in Appearance-Guided Omnidirectional Visual Odometry by Using Vocabulary Trees.” In: *Robotics and Autonomous Systems* 58.6 (2010), pp. 820–827 (↗ pp. 26, 35, 39, 71, 72, 87).

[569] D. Scaramuzza, F. Fraundorfer, and R. Siegwart. “Real-Time Monocular Visual Odometry for On-Road Vehicles with 1-Point RANSAC.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 09).* 2009, pp. 4293–4299 (↗ pp. 35, 71, 72).

[570] D. Scaramuzza, R. Siegwart, and A. Martinelli. “A Robust Descriptor for Tracking Vertical Lines in Omnidirectional Images and its Use in Mobile Robotics.” In: *International Journal of Robotics Research* 28.2 (2009), pp. 149–171 (↗ p. 35).

[571] T. Schairer, B. Huhle, and W. Strasser. “Increased Accuracy Orientation Estimation from Omnidirectional Images Using the Spherical Fourier Transform.” In: *Proceedings of the 3DTV Conference.* May 2009, pp. 1–4 (↗ p. 45).

[572] T. Schairer, B. Huhle, and W. Strasser. “Application of Particle Filters to Vision-Based Orientation Estimation Using Harmonic Analysis.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 10).* May 2010, pp. 2556–2561 (↗ p. 45).

[573] S. Schiffer, T. Baumgartner, and G. Lakemeyer. “A Modular Approach to Gesture Recognition for Interaction with a Domestic Service Robot.” In: *Intelligent Robotics and Applications*. Ed. by S. Jeschke, H. Liu, and D. Schilberg. Vol. 7102. Lecture Notes in Computer Science (LNCS). Springer, 2011, pp. 348–357 (↗ pp. 29, 224).

[574] D. Schleicher, L. M. Bergasa, M. Ocãna, R. Barea, and E. López. “Real-Time Hierarchical Stereo Visual SLAM in Large-Scale Environments.” In: *Robotics and Autonomous Systems* 58.1 (2010), pp. 991–1002 (↗ p. 67).

[575] O. Schlüter. “Erkennung des Loop-Closure-Problems mit Hilfe von Bildvergleichen.” Bachelor’s Thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2010. In German. (↗ pp. 117, 137, 169, 181).

[576] C. Schmid and R. Mohr. “Local Grayvalue Invariants for Image Retrieval.” In: *IEEE Transactions on Pattern Analysis and Machine Vision* 19.5 (1997), pp. 530–535 (↗ p. 59).

[577] A. Schmidt, M. Kraft, M. Fularz, and Z. Domagała. “Comparative Assessment of Point Feature Detectors and Descriptors in the Context of Robot Navigation.” In: *Journal of Automation, Mobile Robotics, and Intelligent Systems* 7.1 (2013), pp. 11–20 (↗ pp. 33, 169).

[578] G. Schnurer. “Saubermänner. 24 Saugroboter von 200 bis 1150 Euro im Test.” In: *c’t Magazin* 11 (2011), pp. 132–145. In German. (↗ p. 12).

[579] R. D. Schraft, U. Bräuning, T. Orlowski, and M. Hornemann. “Automated Cleaning of Windows on Standard Facades.” In: *Automation in Construction* 9.5 (2000), pp. 489–501 (↗ p. 10).

[580] R. D. Schraft and G. Schmierer. *Service Robots*. Peters, 2000 (↗ pp. 9, 10).

[581] C. Schröter, A. König, H. J. Böhme, and H.-M. Gross. “Multi-Sensor Monte-Carlo Localization Combining Omni-Vision and Sonar-Range Sensors.” In: *Proceedings of the European Conference on Mobile Robotics (ECMR 05)*. 2005, pp. 164–169 (↗ pp. 78, 89).

[582] H. Shatkay and L. P. Kaelbling. “Learning Topological Maps with Weak Local Odometric Information.” In: *Proceedings of the Fifteenth International Joint Conference on Artifical Intelligence (IJCAI 97)*. Morgan Kaufmann, 1997, pp. 920–927 (↗ p. 67).

[583] J. Shi and C. Tomasi. “Good Features to Track.” In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 94)*. 1994, pp. 593–600 (↗ p. 35).

[584] H. Y. Shum, S. C. Chan, and S. B. Kang. *Image-Based Rendering*. Springer, 2007 (↗ p. 66).

[585] C. Siagian and L. Itti. “Rapid Biologically-Inspired Scene Classification Using Features Shared with Visual Attention.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.2 (2007), pp. 300–312 (↗ p. 33).

[586] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza. *Introduction to Autonomous Mobile Robots*. 2nd ed. MIT Press, 2011 (↗ pp. 2, 5, 12, 15, 17, 22–24, 33–36, 40, 64–66, 70, 72, 75, 78, 82, 85, 106, 113, 169, 217).

[587] S. Siggelkow and H. Burkhardt. “Fast Invariant Feature Extraction for Image Retrieval.” In: *State-of-the-Art in Content-Based Image and Video Retrieval*. Ed. by R. C. Veltkamp, H. Burkhardt, and H.-P. Kriegel. Computational Imaging and Vision. Kluwer Academic Press, 2001, pp. 43–68 (↗ p. 32).

[588] E. P. Simoncelli and B. Simoncelli. “Natural Image Statistics and Neural Representation.” In: *Annual Reviews of Neuroscience* 24 (2001), pp. 1193–1216 (↗ pp. 122, 129, 174).

[589] M. Simoncelli, G. Zunino, H. I. Christensen, and K. Lange. “Autonomous Pool Cleaning: Self Localization and Autonomous Navigation for Cleaning.” In: *Autonomous Robots* 9.3 (2000), pp. 261–270 (↗ p. 10).

[590] G. Singh and J. Košecká. “Visual Loop Closing Using Gist Descriptors in Manhattan World.” In: *Proceedings of the Workshop on Omnidirectional Robot Vision*. 2010 (↗ pp. 32, 33, 36).

[591] S. P. Singh, A. Verma, and A. K. Shrivastava. “Design and Development of Robotic Sewer Inspection Equipment Controlled by Embedded Systems.” In: *Proceedings of the IEEE International Conference on Emerging Trends in Engineering and Technology (ICETET 08)*. 2008, pp. 1317–1320 (↗ p. 10).

[592] L. Smith, A. Philippides, P. Graham, B. Baddeley, and P. Husbands. “Linked Local Navigation for Visual Route Guidance.” In: *Adaptive Behavior* 15.3 (2007), pp. 257–271 (↗ pp. 54, 83).

[593] L. Smith, A. Philippides, P. Graham, and P. Husbands. “Linked Local Visual Navigation and Robustness to Motor Noise and Route Displacement.” In: *From Animals to Animats*. 2008, pp. 179–188 (↗ pp. 54, 83).

[594] L. Smith, A. Philippides, and P. Husbands. “Navigation in Large-Scale Environments Using an Augmented Model of Visual Homing.” In: *From Animals to Animats*. Ed. by S. Nolfi, G. Baldassarre, R. Calabretta, J. Hallam, D. Marocco, J.-A. Meyer, O. Miglino, and D. Parisi. Vol. 4095. Lecture Notes in Computer Science (LNCS). Springer, 2006, pp. 251–262 (↗ p. 83).

[595] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman. “The New College Vision and Laser Data Set.” In: *International Journal of Robotics Research* 28.5 (2009), pp. 595–599 (↗ p. 88).

[596] M. V. Srinivasan. “Honeybees as a Model for the Study of Visually Guided Flight, Navigation, and Biologically Inspired Robotics.” In: *Physiological Reviews* 91.2 (2011), pp. 413–460 (↗ pp. 42, 56, 60, 66, 86).

[597] M. V. Srinivasan. “Visual Control of Navigation in Insects and its Relevance for Robotics.” In: *Current Opinion in Neurobiology* 21.4 (2011), pp. 535–543 (↗ pp. 42, 56, 60, 66, 86).

[598] M. V. Srinivasan. “Animal Navigation: Ants Match as They March.” In: *Nature* 392.6677 (1998), pp. 660–661 (↗ p. 86).

[599] O. B. Stabell. “Homing and Olfaction in Salmonids: A Critical Review With Special Reference to the Atlantic Salmon.” In: *Biological Reviews* 59.3 (1984), pp. 333–388 (↗ p. 61).

[600] Stemmer Imaging. *Das Handbuch der Bildverarbeitung 2013/2014*. Stemmer Imaging GmbH, 2012. In German. (↗ pp. 25, 217).

[601] A. Štimec, M. Jogan, and A. Leonardis. “Unsupervised Learning of a Hierarchy of Topological Maps Using Omnidirectional Images.” In: *International Journal of Pattern Recognition and Artificial Intelligence* 22.04 (2008), pp. 639–665 (↗ pp. 32, 33, 39, 68, 77).

[602] B. Stroustrup. *The C++ Programming Language*. 3rd ed. Addison-Wesley Longman, 2000 (↗ p. 137).

[603] P. Sturm, S. Ramalingam, J.-P. Tardif, S. Gasparini, and J. Barreto. “Camera Models and Fundamental Concepts Used in Geometric Computer Vision.” In: *Foundations and Trends in Computer Graphics and Vision* 6.1–2 (2010), pp. 1–183 (↗ p. 24).

[604] W. Stürzl and R. Möller. “An Insect-Inspired Active Vision Approach for Orientation Estimation with Panoramic Images.” In: *Proceedings of the International Work-Conference on the Interplay Between Natural and Artificial Computation (IWINAC 07)*. Ed. by J. R. Á. José Mira. Vol. 4527. Lecture Notes in Computer Science (LNCS). Springer, 2007, pp. 61–70 (↗ p. 45).

[605] W. Stürzl, D. Soccol, J. Zeil, N. Böddeker, and M. V. Srinivasan. “Rugged, Obstruction-Free, Mirror-Lens Combination for Panoramic Imaging.” In: *Applied Optics* 47.32 (2008), pp. 6070–6078 (↗ pp. 26, 43, 52, 61, 217).

[606] W. Stürzl and M. V. Srinivasan. “Omnidirectional Imaging System with Constant Elevational Gain and Single Viewpoint.” In: *Proceedings of the Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS 10)*. 2010 (↗ p. 26).

[607] W. Stürzl, M. Suppa, and D. Burschka. “Leight-Weight Panoramic Mirror Design for Visual Navigation.” In: *Workshop Proceedings on the International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR 08)*. 2008, pp. 218–229 (↗ pp. 26, 217).

[608] W. Stürzl, H.-J. Dahmen, and H. A. Mallot. “The Quality of Catadioptric Imaging — Application to Omnidirectional Stereo.” In: *Proceedings of the European Conference on Computer Vision*. Ed. by T. Pajdla and J. Matas. Vol. 3021. Lecture Notes in Computer Science (LNCS). Springer, 2004, pp. 614–627 (↗ p. 26).

[609] W. Stürzl and H. A. Mallot. “Fast Phase-Based Orientation Estimation for Panoramic Images.” In: *Proceedings of the Workshop Dynamic Perception*. Ed. by R. P. Würtz and M. Lappe. Akademische Verlags-Gesellschaft Aka, 2002, pp. 289–294 (↗ pp. 32, 45).

[610] W. Stürzl and H. A. Mallot. “Vision-Based Homing with a Panoramic Stereo Sensor.” In: *Biologically Motivated Computer Vision*. Ed. by H. H. Bülthoff, S.-W. Lee, T. A. Poggio, and C. Wallraven. Vol. 2525. Lecture Notes in Computer Science (LNCS). Springer, 2002, pp. 620–628 (↗ pp. 26, 45, 47, 50, 51, 129, 130).

[611] W. Stürzl and H. A. Mallot. “Efficient Visual Homing Based on Fourier Transformed Panoramic Images.” In: *Robotics and Autonomous Systems* 54.4 (2006), pp. 300–313 (↗ pp. 32, 51, 53, 129, 174, 197, 245).

[612] W. Stürzl and J. Zeil. “Depth, Contrast and View-Based Homing in Outdoor Scenes.” In: *Biological Cybernetics* 96.5 (2007), pp. 519–531 (↗ pp. 30, 45, 52, 59, 123).

[613] R. Swaminathan, M. D. Grossberg, and S. K. Nayar. “Non-Single Viewpoint Catadioptric Cameras: Geometry and Analysis.” In: *International Journal of Computer Vision* 66.3 (2006), pp. 211–229 (↗ p. 24).

[614] R. Swaminathan, S. K. Nayar, and M. D. Grossberg. “Caustics of Catadioptric Cameras.” In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV 01)*. 2001, pp. 2–9 (↗ p. 24).

[615] J. A. Swets, R. M. Dawes, and J. Monahan. “Better Decisions Through Science.” In: *Scientific American* 283.4 (2000), pp. 82–87 (↗ p. 138).

[616] R. Szeliski. “Image-Based Rendering.” In: *Computer Vision Algorithms and Applications*. Springer, 2011, pp. 543–573 (↗ p. 66).

[617] M. D. Szenher. “Visual Homing in Dynamic Indoor Environments.” PhD thesis. Institute of Perception, Action and Behavior, School of Informatics, University of Edinburgh, United Kingdom, 2008 (↗ pp. 52, 128, 155).

[618] M. Szenher. “Visual Homing in Natural Environments.” In: *Proceedings of the Towards Autonomous Robotic Systems (TAROS 05)*. 2005, pp. 221–226 (↗ p. 52).

[619] H. Tamimi, H. Andreasson, A. Treptow, T. Duckett, and A. Zell. “Localization of Mobile Robots with Omnidirectional Vision Using Particle Filter and Iterative SIFT.” In: *Robotics and Autonomous Systems* 54.9 (2006), pp. 758–765 (↗ pp. 26, 35, 39, 78).

[620] L. Tang and S. Yuta. “Indoor Navigation for Mobile Robots Using Memorized Omnidirectional Images and Robot’s Motion.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 02)*. 2002, pp. 269–274 (↗ p. 83).

[621] A. Tapus and R. Siegwart. “Incremental Robot Mapping with Fingerprints of Places.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 05)*. 2005, pp. 2429–2434 (↗ pp. 31–33, 39, 81).

[622] A. Tapus and R. Siegwart. “A Cognitive Modeling of Space Using Fingerprints of Places for Mobile Robot Navigation.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 06)*. 2006, pp. 1188–1193 (↗ pp. 31–33, 39, 81).

[623] A. Tapus and R. Siegwart. “Topological SLAM.” In: *Probabilistic Reasoning and Decision Making in Sensory-Motor Systems*. Ed. by P. Bessière, C. Laugier, and R. Siegwart. Vol. 46. Springer Tracts in Advanced Robotics (STAR). Springer, 2008, pp. 99–127 (↗ pp. 31–33, 39, 81).

[624] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis. “Monocular Visual Odometry in Urban Environments Using an Omnidirectional Camera.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 08)*. 2008, pp. 2531–2538 (↗ pp. 36, 71, 72).

[625] J. Tautz. *The Buzz About Bees: Biology of a Superorganism*. Springer, 2008 (↗ p. 86).

[626] J. B. Tenenbaum, V. Silva, and J. C. Langford. “A Global Geometric Framework for Nonlinear Dimensionality Reduction.” In: *Science* 290.5500 (2000), pp. 2319–2323 (↗ p. 212).

[627] P. Thévenaz and M. Unser. “Optimization of Mutual Information for Multiresolution Image Registration.” In: *IEEE Transactions on Image Processing* 9.12 (2000), pp. 2083–2099 (↗ pp. 128, 155).

[628] S. Thompson and A. Zelinsky. “Accurate Local Positioning Using Visual Landmarks from a Panoramic Sensor.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 02)*. 2002, pp. 2656–2661 (↗ p. 68).

[629] K. Thorup, R. A. Holland, A. P. Tøttrup, and M. Wikelski. “Understanding the Migratory Orientation Program of Birds: Extending Laboratory Studies to Study Free-Flying Migrants in a Natural Setting.” In: *Integrative and Comparative Biology* 50.3 (2010), pp. 315–322 (↗ p. 61).

[630] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005 (↗ pp. 5, 15–17, 22, 39, 40, 64–66, 85, 113).

[631] S. Thrun and J. J. Leonard. “Simultaneous Localization and Mapping.” In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer, 2008, pp. 871–889 (↗ pp. 5, 17, 22, 39, 40, 64–66).

[632] E. Todt and C. Torras. “Detecting Salient Cues Through Illumination-Invariant Color Ratios.” In: *Robotics and Autonomous Systems* 48.2–3 (2004), pp. 111–130 (↗ p. 60).

[633] K. D. Toennies. “Detection and Segmentation by Shape and Appearance.” In: *Guide to Medical Image Analysis*. Advances in Computer Vision and Pattern Recognition (ACVPR). Springer, 2012, pp. 333–378 (↗ p. 20).

[634] E. C. Tolman. “Cognitive Maps in Rats and Men.” In: *Psycological Review* 55 (4 1948), pp. 189–208 (↗ p. 66).

[635] F. Tombari, L. D. Stefano, S. Mattoccia, and A. Galanti. “Performance Evaluation of Robust Matching Measures.” In: *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP 08)*. 2008, pp. 473–478 (↗ pp. 22, 30, 33, 42, 45, 119, 123).

[636] F. Tong, S. K. Tso, and T. Z. Xu. “A High Precision Ultrasonic Docking System Used for Automatic Guided Vehicle.” In: *Sensors and Actuators A: Physical* 118.2 (2005), pp. 183–189 (↗ p. 10).

[637] W. F. Towne. “Honeybees Can Learn the Relationship Between the Solar Ephemeris and a Newly-Experienced Landscape.” In: *Journal of Experimental Biology* 211.23 (2008), pp. 3737–3743 (↗ p. 47).

[638] W. F. Towne and H. Moscrip. “The Connection Between Landscapes and the Solar Ephemeris in Honeybees.” In: *Journal of Experimental Biology* 211.23 (2008), pp. 3729–3736 (↗ p. 47).

[639] B. Tribelhorn and Z. Dodds. “Evaluating the Roomba: A Low-Cost, Ubiquitous Platform for Robotics Research and Education.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 07)*. 2007, pp. 1393–1399 (↗ p. 12).

[640] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. “Bundle Adjustment — A Modern Synthesis.” In: *Proceedings of the International Workshop on Vision Algorithms*. 2000, pp. 298–372 (↗ pp. 39, 65).

[641] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998 (↗ pp. 3, 24, 28, 48, 54, 219, 223).

[642] O. Trullier, S. Wiener, A. Berthoz, and J.-A. Meyer. “Biologically-Based Artificial Navigation Systems: Review and Prospects.” In: *Progress in Neurobiology* 51.5 (1997), pp. 483–544 (↗ pp. 3, 17, 20, 47, 50, 60, 83).

[643] K. Tsubata, K. Suzuki, S. Mikami, and E. I. Osawa. “Recognition of Lawn Information for Mowing Robots.” In: *Proceedings of the IEEE International Conference on Autonomous Robots and Agents (ICARA 09)*. 2009, pp. 15–20 (↗ p. 10).

[644] T. Tuytelaars and K. Mikolajczyk. “Local Invariant Feature Detectors: A Survey.” In: *Foundations and Trends in Computer Graphics and Vision* 3.3 (2007), pp. 177–280 (↗ pp. 33–35, 121, 169, 211).

[645] H. Ueda. “Physiological Mechanism of Homing Migration in Pacific Salmon. From Behavioral to Molecular-Biological Approaches.” In: *General and Comparative Endocrinology* 170.2 (2011), pp. 222–232 (↗ p. 61).

[646] S. Ujvari and O. P. Hilmola. “Simulation of Automatic Guided Vehicle Systems in Manufacturing Environment. Case: Volvo’s Crankshaft Unit in Skovde.” In: *International Journal of Manufacturing Technology and Management* 15.1 (2008), pp. 45–63 (↗ p. 10).

[647] I. Ulrich and I. Nourbakhsh. “Appearance-Based Place Recognition for Topological Localization.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 00)*. 2000, pp. 1023–1029 (↗ pp. 32, 82, 170, 172).

[648] K. Usher, P. Corke, and P. Ridley. “Landmark-Based Nonholonomic Visual Homing.” In: *Field and Service Robotics. Recent Advances in Reserch and Applications.* Ed. by S. Yuta, H. Asama, E. Prassler, T. Tsubouchi, and S. Thrun. Vol. 24. Springer Tracts in Advanced Robotics (STAR). Springer, 2006, pp. 61–70 (↗ p. 54).

[649] K. Usher, P. Ridley, and P. Corke. “Visual Servoing of a Car-Like Vehicle. An Application of Omnidirectional Vision.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA03).* 2003, pp. 4288–4293 (↗ p. 54).

[650] I. Vajk, J. Hetthéssy, and R. Bars. “Control Theory for Automation: Advanced Techniques.” In: *Handbook of Automation.* Ed. by S. Y. Nof. Springer, 2009, pp. 173–198 (↗ pp. 49, 58).

[651] C. Valgren, T. Duckett, and A. Lilienthal. “Incremental Spectral Clustering and its Application to Topological Mapping.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 07).* 2007, pp. 4283–4288 (↗ pp. 35, 36, 81, 82, 88, 89).

[652] C. Valgren and A. Lilienthal. “Incremental Spectral Clustering and Seasons: Appearance-Based Localization in Outdoor Environments.” In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 08).* 2008, pp. 1856–1861 (↗ pp. 35, 36, 81, 88, 89).

[653] C. Valgren, A. J. Lilienthal, and T. Duckett. “Incremental Topological Mapping Using Omnidirectional Vision.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 06).* 2006, pp. 3441–3447 (↗ pp. 26, 35, 36, 81).

[654] C. Valgren and A. J. Lilienthal. “SIFT, SURF and Seasons: Long-Term Outdoor Localization Using Local Features.” In: *Proceedings of the European Conference on Mobile Robotics (ECMR 07).* 2007, pp. 253–258 (↗ pp. 35, 36, 88, 89).

[655] C. Valgren and A. J. Lilienthal. “SIFT, SURF and Seasons: Appearance-Based Long-Term Localization in Outdoor Environments.” In: *Robotics and Autonomous Systems* 58.2 (2010), pp. 157–165 (↗ pp. 26, 35, 36, 82, 87–89).

[656] I. Vallivaara, J. Haverinen, A. Kemppainen, and J. Röning. “Simultaneous Localization and Mapping Using Ambient Magnetic Field.” In: *Proceedings of the IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 10).* 2010, pp. 14–19 (↗ p. 16).

[657] I. Vallivaara, J. Haverinen, A. Kemppainen, and J. Röning. “Magnetic Field-Based SLAM Method for Solving the Localization Problem in Mobile Robot Floor-Cleaning Task.” In: *Proceedings of the IEEE International Conference on Advanced Robotics (ICAR 11).* 2011, pp. 198–203 (↗ p. 16).

[658] A. Vardy. “Long-Range Visual Homing.” In: *Proceedings of the IEEE International Conference on Robotics and Biomimetics.* 2006, pp. 220–226 (↗ p. 82).

[659] A. Vardy. “Using Feature Scale Change for Robot Localization Along a Route.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 10).* 2010, pp. 4830–4835 (↗ pp. 35, 83).

[660] A. Vardy and R. Möller. “Biologically Plausible Visual Homing Methods Based on Optical Flow Techniques.” In: *Connection Science* 17.1-2 (2005), pp. 47–89 (↗ pp. 50, 55, 57–59, 88, 142).

[661] A. Vardy and F. Oppacher. “A Scale Invariant Local Image Descriptor for Visual Homing.” In: *Biomimetic Neural Learning for Intelligent Robots.* Ed. by G. Palm and S. Wermter. Vol. 3575. Lecture Notes in Artificial Intelligence (LNAI). Springer, 2005, pp. 362–381 (↗ p. 58).

[662] A. Vardy. “A Simple Visual Compass with Learned Pixel Weights.” In: *Proceedings of the Canadian Conference on Electrical and Computer Engineering.* May 2008, pp. 1581–1586 (↗ p. 45).

[663] A. Vardy and F. Oppacher. “Low-Level Visual Homing.” In: *Proceedings of the 7th European Conference on Artificial Life (ECAL 07).* Ed. by W. Banzhaf, J. Ziegler, T. Christaller, P. Dietrich, and J. T. Kim. Vol. 2801. Lecture Notes in Computer Science (LNCS). Springer, 2003, pp. 875–884 (↗ p. 55).

[664] A. Vardy and F. Oppacher. “Anatomy and Physiology of an Artificial Vision Matrix.” In: *Biologically Inspired Approaches to Advanced Information Technology (BioADIT 04).* Ed. by A. J. Ijspeert, M. Murata, and N. Wakamiya. Vol. 3141. Lecture Notes in Computer Science (LNCS). Springer, 2004 (↗ p. 55).

[665] A. Vardy and F. Oppacher. “A Scale Invariant Neural Feature Detector for Visual Homing.” In: *Proceedings of KI 04 Workshop on Neurobotics.* Ed. by S. Wermter and G. Palm. 2005 (↗ p. 58).

[666] R. F. Vassallo, J. Santos-Victor, and H. J. Schneebeli. “Using Motor Representations for Topological Mapping and Navigation.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 02).* 2002, pp. 478–483 (↗ pp. 81, 84).

[667] D. Venjakob. “Verbesserung des Blockmatching-Verfahrens durch Feature-Transformationen.” Bachelor’s thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2007. In German. (↗ pp. 17, 50, 59).

[668] D. Venjakob. “Verbesserung eines Roboter-Trackingsystems.” Project thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2009. In German. (↗ pp. 169, 203).

[669] D. Venjakob. “Flusslinien-Matching mit beliebig ausgerichteten Bildern.” Master’s Thesis. Computer Engineering Group, Faculty of Technology, Bielefeld University, Germany, 2010. In German. (↗ pp. 17, 50, 59).

[670] R. Vidal. “Segmentation of Dynamic Scenes Taken by a Moving Central Panoramic Camera.” In: *Imaging Beyond the Pinhole Camera*. Ed. by K. Daniilidis and R. Klette. Vol. 33. Computational Imaging. Springer, 2006, pp. 125–142 (↗ pp. 29, 223).

[671] P. Viola and M. Jones. “Robust Real-Time Object Detection.” In: *International Journal of Computer Vision* 57.2 (2001), pp. 137–154 (↗ p. 212).

[672] I. F. A. Vis. “Survey of Research in the Design and Control of Automated Guided Vehicle Systems.” In: *European Journal of Operational Research* 170.3 (2006), pp. 677–709 (↗ p. 10).

[673] D. Wagner, A. Mulloni, T. Langlotz, and D. Schmalstieg. “Real-Time Panoramic Mapping and Tracking on Mobile Phones.” In: *Proceedings of the IEEE Virtual Reality Conference (VR 10)*. 2010, pp. 211–218 (↗ p. 85).

[674] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg. “Real-Time Detection and Tracking for Augmented Reality on Mobile Phones.” In: *IEEE Transactions on Visualization and Computer Graphics* 16.3 (2010), pp. 355–368 (↗ p. 85).

[675] M. R. Walter, R. M. Eustice, and J. J. Leonard. “Exactly Sparse Extended Information Filters for Feature-Based SLAM.” In: *International Journal of Robotics Research* 26.4 (2007), pp. 335–359 (↗ p. 39).

[676] Z. Wang, S. Huang, and G. Dissanayake. “Tradeoffs in SLAM with Sparse Information Filters.” In: *Field and Service Robotics*. Ed. by C. Laugier and R. Siegwart. Vol. 42. Springer Tracts in Advanced Robotics (STAR). Springer, 2008, pp. 339–348 (↗ pp. 39, 40).

[677] B. Webb. “What Does Robotics Offer Animal Behaviour?” In: *Animal Behavior* 60.5 (2000), pp. 545–558 (↗ pp. 42, 60, 66, 213, 216).

[678] B. Webb. “Can Robots Make Good Models of Biological Behaviour?” In: *Behavioural and Brain Sciences* 24.6 (2001), pp. 1033–1050 (↗ pp. 42, 60, 66, 213, 216).

[679] R. Wehner. “Desert Ant Navigation: How Miniature Brains Solve Complex Tasks.” In: *Journal of Comparative Physiology A* 189.8 (2003), pp. 579–588 (↗ p. 60).

[680] R. Wehner and F. Räber. “Visual Spatial Memory in Desert Ants, Cataglyphis bicolor (Hymenoptera: Formicidae).” In: *Experientia* 35 (1979), pp. 1569–1571 (↗ pp. 42, 47, 60).

[681] R. Wehner, M. Boyer, F. Loertscher, S. Sommer, and U. Menzi. “Ant Navigation: One-Way Routes Rather Than Maps.” In: *Current Biology* 16.1 (2006), pp. 75–79 (↗ p. 86).

[682] R. Wei, D. Austin, and R. Mahony. “Biomimetic Application of Desert Ant Visual Navigation for Mobile Robot Docking with Weighted Landmarks.” In: *International Journal of Intelligent Systems, Technologies, and Applications* 1.1/2 (July 2005), pp. 174–190 (↗ p. 54).

[683] R. Wei, R. Mahony, and D. Austin. “A Bearing-Only Control Law for Stable Docking of Unicycles.” In: *Proceedings of the IEEE/RSJ International Conference on Robots and Systems (IROS 03)*. 2003, pp. 3793–3798 (↗ p. 54).

[684] M. Weiss-Cohen, I. Sirotin, and E. Rave. “Lawn Mowing System for Known Areas.” In: *Proceedings of the IEEE International Conference on Computational Intelligence for Modelling Control and Automation (CIMCA 08)*. 2008, pp. 539–544 (↗ p. 10).

[685] F. Werner, F. Maire, and J. Sitte. “Topological SLAM Using Fast Vision Techniques.” In: *Advances in Robotics*. Ed. by J.-H. Kim, S. Ge, P. Vadakkepat, N. Jesse, A. Al Manum, S. Puthusserypady K, U. Rückert, J. Sitte, U. Witkowski, R. Nakatsu, T. Braunl, J. Baltes, J. Anderson, C.-C. Wong, I. Verner, and D. Ahlgren. Vol. 5744. Lecture Notes in Computer Science (LNCS). Springer, 2009, pp. 187–196 (↗ p. 32).

[686] F. Werner, J. Sitte, and F. Maire. “On the Induction of Topological Maps from Sequences of Colour Histograms.” In: *Proceedings of the Conference of the Australian Pattern Recognition Society on Digital Image Computing Techniques and Applications (DICTA 07)*. 2007, pp. 167–174 (↗ pp. 32, 81).

[687] F. Werner, J. Sitte, and F. Maire. “Topological Map Induction Using Neighbourhood Information of Places.” In: *Autonomous Robots* 32.4 (2012), pp. 405–418 (↗ pp. 31, 32, 81).

[688] F. Werner, J. Sitte, and F. D. Maire. “Automatic Place Determination Using Colour Histograms and Self-Organising Maps.” In: *Proceedings of International Conference on Advanced Robotics (ICAR 07)*. 2007, pp. 111–115 (↗ pp. 31, 32, 82, 170, 172).

[689] D. Weyns, T. Holvoet, K. Schelfthout, and J. Wielemans. “Decentralized Control of Automatic Guided Vehicles: Applying Multi-Agent Systems in Practice.” In: *Proceedings of the ACM SIGPLAN Conference on Object-Oriented Programming Systems Languages and Applications (OOPSLA 08)*. 2008, pp. 663–674 (↗ p. 10).

[690] J. M. Wiener, S. Büchner, and C. Hölscher. “Taxonomy of Human Wayfinding: A Knowledge-Based Approach.” In: *Spatial Cognition and Computation* 9.2 (2009), pp. 152–165 (↗ pp. 61, 86).

[691] J. M. Wiener, S. Shettleworth, V. P. Bingman, K. Cheng, S. Healy, L. F. Jacobs, K. Jeffery, H. A. Mallot, R. Menzel, and N. S. Newcombe. “Animal Navigation. A Synthesis.” In: *Animal Thinking. Contemporary Issues in Comparative Cognition*. Ed. by R. Menzel and J. Fischer. MIT Press, 2011, pp. 52–76 (↗ p. 61).

[692] R. Wiltschko, K. Stapput, P. Thalau, and W. Wiltschko. "Directional Orientation of Birds by the Magnetic Field Under Different Light Conditions." In: *Journal of the Royal Scociety Interface* 7.2 (2009), S163–S177 (↗ pp. 47, 61).

[693] R. Wiltschko and W. Wiltschko. "Avian Navigation." In: *The Auk. An International Journal of Ornithology* 126.4 (2009), pp. 717–743 (↗ pp. 47, 61).

[694] W. Wiltschko and R. Wiltschko. "Global Navigation in Migratory Birds: Tracks, Strategies, and Interactions Between Mechanisms." In: *Current Opinion in Neurobiology* 22.2 (2012), pp. 328–325 (↗ p. 47).

[695] WinterGreen Research. *Cleaning Robot Market Strategies, Shares and Forecasts.* WinterGreen Research, Lexington, Massachusetts, USA, 2010 (↗ p. 1).

[696] B. Q. Wu, W. Sun, and M. H. Ouyang. "Duct Cleaning Robot Comprehensive Kinematics Modeling and Analysis Considering the Complex Terrain." In: *Advanced Materials Research* 430–432.1 (2012), pp. 2032–2036 (↗ p. 10).

[697] O. Wulf, A. Nüchter, J. Hertzberg, and B. Wagner. "Benchmarking Urban Six-Degree-of-Freedom Simultaneous Localization and Mapping." In: *Journal of Field Robotics* 25.3 (2008), pp. 148–163 (↗ p. 88).

[698] G. Wyeth and M. Milford. "Spatial Cognition for Robots." In: *IEEE Robotics and Automation Magazine* 16.3 (2009), pp. 24–32 (↗ pp. 30, 77, 78, 89).

[699] A. Wystrach, G. Beugnon, and K. Cheng. "Ants Might Use Different View-Matching Strategies On and Off the Route." In: *Journal of Experimental Biology* 215.1 (2012), pp. 44–55 (↗ p. 86).

[700] A. Wystrach, S. Schwarz, P. Schultheiss, G. Beugnon, and K. Cheng. "Views, Landmarks, and Routes: How Do Desert Ants Negotiate an Obstacle Course?" In: *Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology* 197.2 (2011), pp. 167–179 (↗ p. 86).

[701] A. Wystrach, G. Beugnon, and K. Cheng. "Landmarks or Panoramas: What do Navigating Ants Attend to for Guidance?" In: *Frontiers in Zoology* 8 (2011), p. 21 (↗ p. 60).

[702] A. Wystrach and P. Graham. "View-Based Matching Can be More Than Image Matching: The Importance of Considering an Animal's Perspective." In: *i-Perception* 3.8 (2012), pp. 547–549 (↗ p. 42).

[703] A. Wystrach and P. Graham. "What Can We Learn from Studies on Insect Navigation." In: *Animal Behavior* 84.1 (July 2012), pp. 13–20 (↗ pp. 42, 43, 60, 83).

[704] Y. Yagi, K. Imai, K. Tsuji, and M. Yachida. "Iconic Memory-Based Omnidirectional Route Panorama Navigation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.1 (Jan. 2005), pp. 78–87 (↗ pp. 82, 83).

[705] Y. Yagi, K. Imai, and M. Yachida. "Iconic Memory-Based Omnidirectional Route Panorama Navigation." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '03).* Sept. 2003, pp. 564–570 (↗ pp. 82, 83).

[706] Y. Yagi, K. Tsuji, and M. Yachida. "Evaluation of Iconic Memory-Based ORP Navigation." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 04).* Vol. 5. May 2004, pp. 4271–4276 (↗ pp. 82, 83).

[707] H. Yaguchi. "Robot Introduction to Cleaning Work in the East Japan Railway Company." In: *Advanced Robotics* 10.4 (1995), pp. 403–414 (↗ p. 10).

[708] M. Yamamoto, Y. Hayashi, K. Nakamura, and A. Mohri. "Development and Motion Analysis of a Cleaner Robot for Vertical Type Air Conditioning Duct." In: *Transactions of the Japan Society of Mechanical Engineers* 17.5 (2005), pp. 1704–1710 (↗ p. 10).

[709] Y. Yamamoto, P. Prijanian, J. Brown, M. Munich, E. Di Bernardo, L. Goncalves, J. Ostrowski, and N. Karlsson. "Optical Sensing for Robot Perception and Localization." In: *Proceedings of the IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO 05).* 2005, pp. 14–17 (↗ p. 16).

[710] B. Yamauchi. "A Frontier-Based Approach for Autonomous Exploration." In: *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA 97).* 1997, pp. 146–151 (↗ p. 15).

[711] B. Yamauchi, A. Schultz, and W. Adams. "Mobile Robot Exploration and Map-Building with Continuous Localization." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 98).* Vol. 4. 1998, pp. 3715–3720 (↗ p. 15).

[712] S. Yi, B. Choi, and N. Ahuja. "Real-Time Omni-Directional Distance Measurement with Active Panoramic Vision." In: *International Journal of Control, Automation, and Systems* 5.2 (2007), pp. 184–191 (↗ p. 223).

[713] W. Yoshizaki, Y. Mochizuki, N. Ohnishi, and A. Imiya. "Free Space Detection From Catadioptric Omnidirectional Images for Visual Navigation Using Optical Flow." In: *Proceedings of the Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras (OMNIVIS 08).* 2008 (↗ pp. 55, 220, 223).

[714] S. E. Yu and D. E. Kim. "Landmark Vectors With Quantized Distance Information for Homing Navigation." In: *Adaptive Behavior* 19.2 (2011), p. 121 (↗ p. 54).

[715] S.-E. Yu, C. Lee, and D. Kim. "Analyzing the Effect of Landmark Vectors in Homing Navigation." In: *Adaptive Behavior* 20.5 (June 2012), pp. 337–359 (↗ p. 54).

[716] M. Zapka, D. Heyers, C. M. Hein, S. Engels, N.-L. Schneider, J. Hans, S. Weiler, D. Dreyer, D. Kishkinev, J. M. Wild, and H. Mouritsen. “Visual but not Trigeminal Mediation of Magnetic Compass Information in a Migratory Bird.” In: *Nature* 461.7268 (2009), p. 1274 (↗ pp. 47, 61).

[717] J. Zeil. “Visual Homing: An Insect Perspective.” In: *Current Opinion in Neurobiology* 22.2 (2012), pp. 285–293 (↗ pp. 42, 47, 56, 60, 66, 86, 212).

[718] J. Zeil, M. I. Hoffmann, and J. S. Chahl. “Catchment Areas of Panoramic Images in Outdoor Scenes.” In: *Journal of the Optical Society of America A* 20.3 (2003), pp. 450–469 (↗ pp. 5, 29, 45–47, 50, 52, 96, 117, 119, 120, 128, 163, 166, 215, 218).

[719] J. Zeil, N. Boeddeker, and W. Stürzl. “Visual Homing of Insects and Robots.” In: *Flying Insects and Robots.* Ed. by D. Floreano, J.-C. Zufferey, M. V. Srinivasan, and C. Ellington. Springer, 2009, pp. 87–100 (↗ pp. 50, 60).

[720] A. M. Zhang and L. Kleeman. “Robust Appearance-Based Visual Route Following for Navigation in Large-Scale Outdoor Environments.” In: *International Journal of Robotics Research* 28.3 (2009), pp. 331–356 (↗ pp. 30, 76, 78, 83, 84).

[721] H. Zhang, J. Zhang, W. Wang, R. Liu, and G. Zong. “A Series of Pneumatic Glass-Wall Cleaning Robots for High-Rise Buildings.” In: *Industrial Robot: An International Journal* 34.2 (2007), pp. 150–160 (↗ p. 10).

[722] Y. Zhang, P. Wang, J. X. Li, S. W. Yin, and J. L. Xin. “Path Segmentation Algorithm for Automatic Guided Vehicle Based on Machine Vision.” In: *Key Engineering Materials* 431–432 (2010), pp. 330–333 (↗ p. 10).

[723] F. Zhou, B. Peng, Y. Cui, Y. Wang, and H. Tan. “A Novel Laser Vision Sensor for Omnidirectional 3D Measurement.” In: *Optics and Laser Technology* 45.1 (2013), pp. 1–12 (↗ p. 223).

[724] P. Zingaretti and E. Frontoni. “Appearance Based Robotics.” In: *IEEE Robotics and Automation Magazine* 13.1 (2006), pp. 59–68 (↗ pp. 32, 39, 78).

[725] Z. Zivkovic, B. Bakker, and B. Kröse. “Hierarchical Map Building Using Visual Landmarks and Geometric Constraints.” In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 05).* 2005, pp. 2480–2485 (↗ pp. 35, 36, 68, 81, 88, 211, 212).

[726] Z. Zivkovic, O. Booij, and B. Kröse. “From Images to Rooms.” In: *Robotics and Autonomous Systems* 55.5 (2007), pp. 411–418 (↗ pp. 35, 68, 81, 88, 211, 212).

[727] D. Zoran and Y. Weiss. “Scale Invariance and Noise in Natural Images.” In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV 09).* 2009, pp. 2209–2216 (↗ p. 186).

Internet Sources

[I1] E. Ackerman. *Bosch Introduces New Autonomous Robotic Lawnmower.* URL: http://spectrum.ieee.org/automaton/robotics/home-robots/bosch-introduces-new-autonomous-robotic-lawnmower (visited on 2012-11-22) (↗ pp. 10, 217).

[I2] Agait. *EC01.* URL: http://www.agaitech.com/user/list.jsp?id1=20&id2=498 (visited on 2012-05-08) (↗ p. 12).

[I3] Agait. *EC01 Enhanced.* URL: http://www.agaitech.com/user/list.jsp?id1=122&id2=659 (visited on 2012-05-08) (↗ p. 12).

[I4] Agait. *EC02.* URL: http://www.agaitech.com/user/list.jsp?id1=123&id2=652 (visited on 2012-05-08) (↗ p. 12).

[I5] Agait. *EC02 Mini.* URL: http://www.agaitech.com/user/list.jsp?id1=125&id2=677 (visited on 2012-05-08) (↗ p. 12).

[I6] Agama. *AiRobot RC320A.* URL: http://www.agamazone.com/products_rc_320a.html (visited on 2012-05-08) (↗ p. 12).

[I7] Agama. *AiRobot RC330A.* URL: http://www.agamazone.com/products_rc_330a.htmlhttp://www.agamazone.com/products_rc_330a.html (visited on 2012-05-08) (↗ p. 12).

[I8] Agama. *AiRobot RC520A.* URL: http://www.agamazone.com/products_rc_520a.html (visited on 2012-05-08) (↗ p. 12).

[I9] Agama. *AiRobot RC530A.* URL: http://www.agamazone.com/products_rc_530a.html (visited on 2012-05-08) (↗ p. 12).

[I10] ALM Robotics. *Windoro Window Cleaning Robot.* URL: http://www.autowindowcleaner.co.uk (visited on 2012-04-16) (↗ p. 10).

[I11] T. Antalóczy. *Robot Buying Guide.* URL: http://robotbg.com/robots/floor_cleaners (visited on 2012-06-01) (↗ p. 12).

[I12] Asus. *Xtion PRO LIVE.* URL: http://www.asus.com/Multimedia/Xtion_PRO_LIVE/ (visited on 2013-10-21) (↗ p. 224).

[I13] Atmel. *ARM Procssor-Based Microcontrollers.* URL: http://www.atmel.com/products/microcontrollers/arm/default.aspx (visited on 2013-01-23) (↗ p. 217).

[I14] Bielefeld Marketing. *GENIALE Bielefeld.* URL: http://www.geniale-bielefeld.de (visited on 2013-01-20). In German. (↗ p. 115).

[I15] Bielefeld University — Computer Engineering Group. *Panoramic Image Databases.* URL: http://www.ti.uni-bielefeld.de/html/research/avardy/index.html (visited on 2012-09-21) (↗ p. 88).

[I16] Bosch Media Service. *Schneller als herkömmliche Geräte: Der Roboter-Mäher Indego von Bosch mäht den Rasen systematisch und in Bahnen.* URL: http://www.bosch-presse.de/presseforum/details.htm?txtID=5813 (visited on 2012-11-22). In German (↗ pp. 10, 217).

[I17] Cyclone Ventilation. *CY-BOT.* URL: http://www.cyclone-industries.com/html/english/cy_bot.html (visited on 2012-04-16) (↗ p. 10).

[I18] Danduct Clean. *Danduct Icetech.* URL: http://www.danduct.com/page.asp?ID=1&ProductID=43 (visited on 2012-04-16) (↗ p. 10).

[I19] Danduct Clean. *Danduct Multi Purpose Robot.* URL: http://www.danduct.com/page.asp?ID=1&ProductID=38 (visited on 2012-04-16) (↗ p. 10).

[I20] Ecovacs. *Winbot.* URL: http://www.ecovacs.com.cn/Winbot/Winbot_data.htm (visited on 2012-04-16) (↗ p. 10).

[I21] egemin Automation. *Automatic Guided Vehicles (AGVs).* URL: http://www.egeminusa.com/pages/agvs/agv_types.html (visited on 2012-04-17) (↗ p. 10).

[I22] eTeks. *Sweet Home 3D.* URL: http://www.sweethome3d.com/ (visited on 2012-07-09) (↗ p. 18).

[I23] Evolution Robotics. *Mint.* URL: http://mintcleaner.com/products/mint-plus/ (visited on 2012-05-08) (↗ p. 12).

[I24] Evolution Robotics. *Mint Plus.* URL: http://mintcleaner.com/products/original-mint/ (visited on 2012-05-08) (↗ p. 12).

[I25] Floorbotics. *IVAC.* URL: http://www.floorbotics.com/ (visited on 2012-04-16) (↗ p. 10).

[I26] Friendly Robotics. *Robomow.* URL: http://www.robomow.com/en-USA/products-usa (visited on 2012-03-26) (↗ p. 10).

[I27] M. Frigo and S. G. Johnson. *FFTW library.* url: http://www.fftw.org/index.html (visited on 2011-09-19) (↗ p. 181).

[I28] Frog AGV Systems. *About Frog AGV Systems.* url: http://www.frog.nl/index.php (visited on 2012-04-17) (↗ p. 10).

[I29] Fuji Heavy Industries. *Guide to FHI | Eco Technologies Company.* url: http://www.fhi.co.jp/english/outline/section/ecotech.html (visited on 2012-04-16) (↗ p. 10).

[I30] Gardena. *Robot mower R40Li.* url: http://www.gardena.com/de/lawn-care/robotic-mower/ (visited on 2012-04-16) (↗ p. 10).

[I31] Google. *Google 3D Warehouse.* url: http://sketchup.google.com/3dwarehouse/ (visited on 2012-07-09) (↗ p. 18).

[I32] Google 3D Warehouse. *6' tall by 10' wide Bookshelf.* url: http://sketchup.google.com/3dwarehouse/details?mid=9e6508b65d8187cad810b14a81e12eca (visited on 2012-07-09) (↗ p. 18).

[I33] Google 3D Warehouse. *Coffee Table.* url: http://sketchup.google.com/3dwarehouse/details?mid=2cfd6534a6a57548d20545a75c46659d (visited on 2012-07-09) (↗ p. 18).

[I34] Google 3D Warehouse. *Ikea Cream Sofa.* url: http://sketchup.google.com/3dwarehouse/details?mid=2f399f006705e8d6e7a01b7619680d4 (visited on 2012-07-09) (↗ p. 18).

[I35] Google 3D Warehouse. *IKEA Uldum carpet black, grey and white.* url: http://sketchup.google.com/3dwarehouse/details?mid=e913c488befe5bea89e08a687ee72988 (visited on 2012-07-09) (↗ p. 18).

[I36] Google 3D Warehouse. *Indoor potted tree.* url: http://sketchup.google.com/3dwarehouse/details?mid=19042f85f93a8b239f7e329438453d00 (visited on 2012-07-09) (↗ p. 18).

[I37] Google 3D Warehouse. *Livingroom computer desk.* url: http://sketchup.google.com/3dwarehouse/details?mid=6ca288da52db2a26128c011f7f6a4a8e (visited on 2012-07-09) (↗ p. 18).

[I38] Google 3D Warehouse. *Potted plant.* url: http://sketchup.google.com/3dwarehouse/details?mid=4d637018815139ab97d540195229f372 (visited on 2012-07-09) (↗ p. 18).

[I39] Google 3D Warehouse. *Quadro6.* url: http://sketchup.google.com/3dwarehouse/details?mid=e75e21c5dff496bd251990397636975f (visited on 2012-07-09) (↗ p. 18).

[I40] Google 3D Warehouse. *Schrankwand.* url: http://sketchup.google.com/3dwarehouse/details?mid=8e40707267d270adb1692c750c823dd9 (visited on 2012-07-09) (↗ p. 18).

[I41] GoPano. *GoPano micro.* url: http://gopanouk.myshopify.com/products/gopano-micro#page=technology (visited on 2012-11-29) (↗ pp. 26, 217).

[I42] P. Graham. *Maps for Bees? (again).* url: http://insectandrobotnavigation.wordpress.com/2012/07/16/why-study-what-we-study/ (visited on 2012-09-25) (↗ p. 86).

[I43] P. Grey. *Point Grey Ladybug 5.* url: http://ww2.ptgrey.com/Products/Ladybug5/ (visited on 2013-11-01) (↗ p. 26).

[I44] D. J. Hill. *360° Panoramic Video Lets You Capture Everything Around You With Your Smartphone.* url: http://singularityhub.com/2012/05/02/360%C3%82%C2%B0-panoramic-video-lets-you-capture-everything-around-you-with-your-smartphone/ (visited on 2012-11-29) (↗ p. 26).

[I45] A. Howard and N. Roy. *RADISH Complete Repository Listing.* url: http://cres.usc.edu/radishrepository/view-all.php (visited on 2012-09-21) (↗ p. 88).

[I46] Husqvarna. *Husqvarna robotic mowers.* url: http://www.husqvarna.com/de/products/robotic-mowers/husqvarna-robotic-mowers-for-homeowners/ (visited on 2012-04-16) (↗ p. 10).

[I47] IBG Hydro Tech. *HydroCut Fräsroboter.* url: http://www.ibg-hydro-tech.de/index.php?option=com_content&view=article&id=52&Itemid=54 (visited on 2012-04-16) (↗ p. 10).

[I48] T. M. R. P. T. (initiative. *Mobile Robot Programming Toolkit Datasets.* url: http://www.mrpt.org/robotic_datasets (visited on 2012-09-21) (↗ p. 88).

[I49] Intel. *Intel Atom Processor.* url: http://www.intel.eu/content/www/eu/en/processors/atom/atom-processor.html (visited on 2013-01-23) (↗ p. 217).

[I50] Intel. *Intel C++ Intrinsic Reference.* url: http://software.intel.com/sites/default/files/m/9/4/c/8/e/18072-347603.pdf (visited on 2013-02-17) (↗ p. 166).

[I51] Intellibot. *AeroBot.* url: http://intellibotrobotics.com/products/aerobot/ (visited on 2012-04-16) (↗ p. 10).

[I52] Intellibot. *DuoBot.* url: http://intellibotrobotics.com/products/duobot/ (visited on 2012-04-16) (↗ p. 10).

[I53] Intellibot. *HydroBot.* url: http://intellibotrobotics.com/products/hydrobot/ (visited on 2012-04-16) (↗ p. 10).

[I54] iRobot. *iRobot Create Programmable Robot.* url: http://www.irobot.com/us/learn/Educators/Create.aspx (visited on 2013-10-11) (↗ p. 2).

[I55] iRobot. *iRobot Verro Pool Cleaning Robots.* url: http://www.irobotverro.com/compare-irobot-verro.html (visited on 2012-04-16) (↗ p. 10).

[I56] iRobot. *Roomba series product page.* url: http://www.irobot.com/de/roomba_range.aspx (visited on 2012-05-08) (↗ p. 12).

[I57] iRobot. *Scooba series product page.* url: http://www.irobot.com/de/scooba_range.aspx (visited on 2012-05-08) (↗ p. 12).

[I58] JBT Corporation. *Automatic Guided Vehicle (AGV) Systems.* url: http://www.jbtc-agv.com/en/solutions/products (visited on 2012-04-17) (↗ p. 10).

[I59] Jervis B.
bibnamedelimb Webb Company. *Products.* url: http://www.jervisbwebb.com/Products.aspx (visited on 2012-04-17) (↗ p. 10).

[I60] Kärcher. *RC3000.* url: http://www.kaercher.de/de/Produkte/Home__Garden/Sauger/Robocleaner/12691010.htm (visited on 2012-05-08) (↗ p. 12).

[I61] Kärcher. *RC4000.* url: http://www.kaercher.de/de/Produkte/Home__Garden/Sauger/Robocleaner/12692000.htm (visited on 2012-05-08) (↗ p. 12).

[I62] kogeto. *kogetodot.* url: http://www.kogeto.com/say-hello-to-dot (visited on 2012-11-29) (↗ pp. 26, 217).

[I63] LG. *Hom-Bot 2.0.* url: http://www.lg.com/de/haushaltsgeraete/bodenpflege/LG-VR6170LVM.jsp (visited on 2012-05-08) (↗ pp. 12, 14).

[I64] LG. *Hom-Bot Square VR6260LVM.* url: http://www.lg.com/de/staubsauger/lg-VR6260LVM (visited on 2012-11-22) (↗ pp. 12, 14).

[I65] Mamirobot. *Cleaning robot product page.* url: http://www.mamirobot.com/ (visited on 2012-05-08) (↗ p. 12).

[I66] Manfrotto. *055XBPROB Pro Tripod.* url: http://www.manfrotto.com/055xprob-pro-tripod-black (visited on 2013-10-21) (↗ p. 203).

[I67] Manfrotto. *Conpact Ball Head with RC2 496RC2.* url: http://www.manfrotto.com/compact-ball-head-with-rc2 (visited on 2013-10-21) (↗ p. 203).

[I68] Manfrotto. *Mini Ball Head with RC2 Rapid Connect System.* url: http://www.manfrotto.com/mini-ball-head-with-rc2-rapid-connect-system (visited on 2013-01-21) (↗ p. 107).

[I69] Microsoft. *Kinect for Windows.* url: http://www.microsoft.com/en-us/kinectforwindows/ (visited on 2013-10-21) (↗ p. 224).

[I70] Moneual. *Rydis R750.* url: http://www.rydis.net/eng/main/main.php (visited on 2012-05-08) (↗ pp. 12, 14).

[I71] M. E. Munich, J.-S. Gutmann, and P. Fong. *Mint: The First Intelligent Mopper and Sweeper Robot.* url: http://www.ifr.org/uploads/media/12_Evolution_robotics.pdf (visited on 2012-11-25) (↗ pp. 1, 2, 11, 12).

[I72] neato robotics. *XV Signature.* url: http://www.neatorobotics.com/product/robotic-vacuums/neato-xv-signature/ (visited on 2013-10-11) (↗ pp. 12, 14).

[I73] neato robotics. *XV Signature Pro.* url: http://www.neatorobotics.com/product/robotic-vacuums/neato-xv-signature-pro/ (visited on 2013-10-11) (↗ pp. 12, 14).

[I74] neato robotics. *XV-15.* url: http://www.neatorobotics.com/eur_en/product/xv-15/ (visited on 2013-10-11) (↗ pp. 12, 14).

[I75] neato robotics. *XV-25.* url: http://www.neatorobotics.com/eur_en/product/xv-25/ (visited on 2013-10-11) (↗ pp. 12, 14).

[I76] O. Pele and M. Werman. *Fast Earth Mover's Distance (EMD) Code.* url: http://www.cs.huji.ac.il/~ofirpele/FastEMD/code/ (visited on 2011-09-19) (↗ p. 181).

[I77] Philips. *HomeRun FC9910.* url: http://www.philips.de/c/staubsauger/homerun-fc9910_01/prd/ (visited on 2012-05-08) (↗ pp. 12, 14).

[I78] A. Pronobis. *COLD. COsy Localization Database.* url: http://www.cas.kth.se/COLD/ (visited on 2012-09-21) (↗ p. 88).

[I79] Rawseeds Project. *Rawseeds Datasets.* url: http://www.rawseeds.org/home/category/benchmarking-toolkit/datasets/ (visited on 2012-09-21) (↗ p. 88).

[I80] robosoft. *AutoVac 6.* url: http://www.robosoft.com/eng/sous_categorie.php?id=1018 (visited on 2012-04-16) (↗ p. 10).

[I81] Samsung. *NaviBot SR-8855.* url: http://www.samsung.com/de/consumer/home-appliances/vacuum-cleaner/vacuum-cleaning-robot/VCR8855L3B/XET (visited on 2012-05-08) (↗ pp. 12, 14, 15).

[I82] Samsung. *NaviBot SR-8855 Silencio.* url: http://www.samsung.com/de/consumer/home-appliances/vacuum-cleaner/vacuum-cleaning-robot/VCR8895L3A/XET (visited on 2012-05-08) (↗ pp. 12, 14, 15).

[I83] D. Scaramuzza. *A Tutorial on Visual Odometry.* url: http://robotics.ethz.ch/~scaramuzza/Davide_Scaramuzza_files/publications/pdf/Visual_Odometry_Tutorial.pdf (visited on 2013-01-30) (↗ p. 24).

[I84] Serbot Innovations. *Gekko Junior G1.* url: http://www.serbot.ch/gekko/gekko-iii-junior/index.php (visited on 2012-04-16) (↗ p. 10).

[I85] Serbot Innovations. *Gekko Junior G3.* url: http://www.serbot.ch/gekko/gekko-junior-g3/index.html (visited on 2012-04-16) (↗ p. 10).

[I86] Serbot Innovations. *Gekko Plus.* url: http://www.serbot.ch/gekko/gekko-iii-plus/index.php (visited on 2012-04-16) (↗ p. 10).

[I87] Serbot Innovations. *Gekko Solar Farm.* url: http://www.serbot.ch/gekko/gekko-solar-farm/index.html (visited on 2012-04-16) (↗ p. 10).

[I88] Solarbrush. *Solarbrush H.* url: http://www.solarbrush.de/home/solarbrush-h (visited on 2012-04-16) (↗ p. 10).

[I89] Solarbrush. *Solarbrush L.* url: http://www.solarbrush.de/home/solarbrush-l (visited on 2012-04-16) (↗ p. 10).

[I90] Sony. *Sony Bloggie MHS-PM5K.* url: http://www.sony.de/hub/mobile-hd-snap-kamera/bloggie-features/article/360-video (visited on 2012-11-29) (↗ pp. 26, 217).

[I91] C. Stachniss. *OpenSLAM.* url: http://www.openslam.org/links.html (visited on 2012-09-21) (↗ p. 88).

[I92] Swiss Federal Institute of Technology — Autonomous Systems Laboratory. *First International Cleaning Robots Contest.* url: http://www.iros02.ethz.ch/contest.php (visited on 2012-03-26) (↗ p. 14).

[I93] Tcl Developer Xchange. *Standard Tcl Library (tcllib).* url: http://tcl.tk/software/tcllib/ (visited on 2011-09-19) (↗ pp. 181, 197).

[I94] Ka-Te. *Ka-Te Grinding Robots.* url: http://www.ka-te.ch/index.php?id=21&L=1 (visited on 2012-04-16) (↗ p. 10).

[I95] Hi-Tech Group. *Intellicart AGV.* url: http://www.hitechroboticsystemz.com/intelliCart.asp (visited on 2012-04-17) (↗ p. 10).

[I96] The Robocup Federation. *RoboCup.* url: http://www.robocup.org/ (visited on 2012-03-25) (↗ p. 14).

[I97] O. an der Universität Bielefeld. *Aktuelle Wetterdaten.* url: http://wetter.osk.uni-bielefeld.de/ (visited on 2011-10-12) (↗ pp. 202, 204, 206).

[I98] University of Oxford — Mobile Robotics Group. *The Oxford Mobile Robotics Group Datasets.* url: http://www.robots.ox.ac.uk/~mobile/wikisite/pmwiki/pmwiki.php?n=Main.Datasets (visited on 2012-09-21) (↗ p. 88).

[I99] Vort Robotics. *Evovacs Deebot D54.* url: http://vort-robotics.de/deebot-roboter-staubsauger/deebot-d54/ (visited on 2012-05-08) (↗ p. 12).

[I100] Vort Robotics. *Evovacs Deebot D56.* url: http://vort-robotics.de/deebot-roboter-staubsauger/deebot-d56/ (visited on 2012-05-08) (↗ p. 12).

[I101] Vort Robotics. *Evovacs Deebot D58.* url: http://vort-robotics.de/deebot-roboter-staubsauger/deebot-d58/ (visited on 2012-05-08) (↗ p. 12).

[I102] Vort Robotics. *Evovacs Deebot D73.* url: http://vort-robotics.de/deebot-roboter-staubsauger/deebot-d73/ (visited on 2012-05-08) (↗ p. 12).

[I103] Vort Robotics. *Evovacs Deebot D76.* url: http://vort-robotics.de/deebot-roboter-staubsauger/deebot-d76/ (visited on 2012-05-08) (↗ p. 12).

[I104] Vort Robotics. *Hanool S100.* url: http://vort-robotics.de/deebot-roboter-staubsauger/ottoro-s100/ (visited on 2012-05-08) (↗ p. 12).

[I105] Vort Robotics. *Hanool S100 Ion.* url: http://vort-robotics.de/deebot-roboter-staubsauger/s100-ion/ (visited on 2012-05-08) (↗ p. 12).

[I106] Vorwerk. *Kobold VR100.* url: http://kobold.vorwerk.com/de/produkte/saugroboter-kobold-vr100/ (visited on 2012-05-08) (↗ pp. 12, 14).

[I107] Z. Winkler. *Cleaning 2002.* url: http://robotika.cz/competitions/cleaning/2002/en (visited on 2012-03-26) (↗ p. 14).

[I108] Youtube user "beerplus18". *Visual SLAM of Robot Vacuum Cleaner (Samsung Hauzen RE70V).* url: http://www.youtube.com/watch?v=bq5HZzGF3vQ (visited on 2013-01-09) (↗ pp. 15, 75).

[I109] Yujin Robot. *iClebo arte.* url: http://www.iclebo.com/english/main_home.html (visited on 2012-11-22) (↗ pp. 12, 14).

[I110] Yujin Robot. *iClebo home.* url: http://www.iclebo.com/english/main_home.html (visited on 2012-11-22) (↗ p. 12).

[I111] Yujin Robot. *Iclebo Kobuki.* url: http://kobuki.yujinrobot.com/home-en/ (visited on 2013-10-11) (↗ p. 2).

[I112] Yujin Robot. *iClebo smart.* url: http://www.iclebo.com/english/main_smart.html (visited on 2012-11-22) (↗ pp. 12, 14).

[I113] Z. Zivkovic. *From Sensors to Human Spatial Concepts.* url: http://www.science.uva.nl/sites/cogniron/fs2hsc/website/ (visited on 2012-09-21) (↗ p. 88).

[I114] Zodiac Pool Systems. *Polaris 9300 sports.* url: http://www.polarispool.com/poolcleaners/9300.aspx (visited on 2012-04-16) (↗ p. 10).

[I115] Zodiac Pool Systems. *Polaris 9300xi.* url: http://www.polarispool.com/poolcleaners/9300xi.aspx (visited on 2012-04-16) (↗ p. 10).

[I116] Zucchetti. *Ambrogio product page.* url: http://www.roboticazucchetti.it/robotica/servlet/pocs_bnav?p_containerAlias=Ambrogio (visited on 2012-04-16) (↗ p. 10).

Datasheets and Technical Specifications

[D1] ACCOWLE. *Omnidirectional Vision Sensor.* URL: http://www.accowle.com/english/products.html (visited on 2012-11-29) (↗ pp. 25, 115).

[D2] ADEPT MOBILEROBOTS. *Pioneer P3-DX.* URL: www.mobilerobots.com/Libraries/Downloads/Pioneer3DX-P3DX-RevA.sflb.ashx (visited on 2012-12-22) (↗ p. 115).

[D3] AXIS COMMUNICATIONS. *Axis 211W Network Camera.* URL: http://www.axis.com/files/datasheet/ds_211w_42335_en_1103_lo.pdf (visited on 2013-01-07) (↗ pp. 107, 203).

[D4] BULLMAN. *BullMan G-KLASSE i7 15hd+/15fh.* URL: http://www.bullman.de/download/dblatt_bullman_g-klassei715.pdf (visited on 2010-02-03) (↗ pp. 104, 201).

[D5] COSMICAR/PENTAX. *1/3 Format Manual Iris Vari-Focal Lens TS2V314A.* URL: http://www.rmaelectronics.com/specsheets/page29.pdf (visited on 2012-11-29) (↗ p. 25).

[D6] DIRECTED PERCEPTION. *Model PTU-D46.* URL: http://www.dperception.com/pdf/specs-ptu-d46.pdf (visited on 2013-01-07) (↗ p. 203).

[D7] IDS IMAGING. *uEye UI-1246-C/M.* URL: http://www.ids-imaging.de/pdfmodule/products_pdf.php?cam_id=160 (visited on 2012-11-29) (↗ p. 25).

[D8] IDS IMAGING. *uEye UI-2220-C/M.* URL: http://www.ids-imaging.de/pdfShow.php?file=frontend/files/2220_D.pdf (visited on 2012-11-29) (↗ pp. 25, 27, 104).

[D9] IDS IMAGING. *uEye UI-2220-C/M.* URL: http://www.ids-imaging.de/pdfShow.php?file=frontend/files/2220_D.pdf (visited on 2009-03-19) (↗ pp. 104, 201).

[D10] IEI TECHNOLOGY CORPORATION. *User Manual revision 1.01.* URL: http://www.icpamerica.com/files/PM-US15W_UMN_V1.01.pdf (visited on 2010-02-01) (↗ pp. 104, 201).

[D11] IMAGINGSOURCE. *DFK 4303.* Email communication. July 2010 (↗ p. 25).

[D12] SHARP. *GP2D12 Optoelectronic Device.* URL: http://www.sharpsma.com/webfm_send/1203 (visited on 2012-11-29) (↗ pp. 104, 223).

[D13] SHARP. *GP2D120 Optoelectronic Device.* URL: http://www.sharpsma.com/webfm_send/1205 (visited on 2012-11-29) (↗ pp. 104, 223).

[D14] SUNEX. *DSL215 Miniature Megapixel Fisheye Lens.* URL: http://www.optics-online.com/OOL/DSL/DSL215.PDF (visited on 2012-11-29) (↗ p. 25).

[D15] TAMRON. *Datenblatt 13VG550ASII.* URL: http://www.tamron.co.jp/en/data/cctv/pdf/spec/v/13VG550ASII.pdf (visited on 2013-01-07) (↗ p. 203).

[D16] TATEYAMA. *PAL Lenses for Surveillance.* Email communication. Sept. 2007 (↗ pp. 25, 27, 104, 201).

Supplemental Materials

[S1] L. Gerstmayr-Hillen. *Image Databases for Loop-Closure Detection Experiments.* url: `http://www.ti.uni-bielefeld.de/downloads/research/topmaps/imagedb.zip` (visited on 2011-10-05) (↗ pp. 88, 135, 181).

[S2] L. Gerstmayr-Hillen, F. Röben, M. Krzykawski, S. Kreft, D. Venjakob, and R. Möller. *Video "Dense Topological Maps and Partial Pose Estimation for Visual Control of an Autonomous Cleaning Robot". Disturbed Odometry Only (Control Experiment).* url: `http://www.ti.uni-bielefeld.de/downloads/research/topmaps/tria_odoonly.mp4` (visited on 2011-12-23) (↗ pp. 103, 106).

[S3] L. Gerstmayr-Hillen, F. Röben, M. Krzykawski, S. Kreft, D. Venjakob, and R. Möller. *Video "Dense Topological Maps and Partial Pose Estimation for Visual Control of an Autonomous Cleaning Robot". Visual Control With Odometry Disturbance.* url: `http://www.ti.uni-bielefeld.de/downloads/research/topmaps/tria_visdisturbed.mp4` (visited on 2011-12-23) (↗ pp. 103, 109).

[S4] L. Gerstmayr-Hillen, F. Röben, M. Krzykawski, S. Kreft, D. Venjakob, and R. Möller. *Video "Dense Topological Maps and Partial Pose Estimation for Visual Control of an Autonomous Cleaning Robot". Visual Control Without Odometry Disturbance.* url: `http://www.ti.uni-bielefeld.de/downloads/research/topmaps/tria_visundisturbed.mp4` (visited on 2011-12-23) (↗ p. 103).

[S5] O. Schlüter, L. Gerstmayr-Hillen, M. Krzykawski, and R. Möller. *Video "Parsimonious Loop-Closure Detection Based on Global Image-Descriptors of Panoramic Images".* url: `http://www.ti.uni-bielefeld.de/downloads/research/topmaps/lcddemo.avi` (visited on 2011-10-12) (↗ p. 202).