Vision-Based Manipulative Gesture Recognition in a Human-Robot Interaction Scenario

Zhe Li

M. Sc. Zhe Li
AG Angewandte Informatik
Technische Fakultät
Universität Bielefeld
email: lizhe@techfak.uni-bielefeld.de

Genehmigte Dissertation zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.).
Von Zhe Li am 23. April 2008
der Technischen Fakultät an der Universität Bielefeld vorgelegt.
Am 10. Juli 2008 verteidigt und genehmigt.

Gutachter:

Prof. Dr. Gerhard Sagerer, Universität Bielefeld
Prof. Dr. Jianwei Zhang, Universität Hamburg
Dr. -Ing. Jannik Fritsch, Honda Research Institute Europe

Prüfungsausschuss:

Prof. Dr. Ipke Wachsmuth, Universität Bielefeld
Prof. Dr. Gerhard Sagerer, Universität Bielefeld
Prof. Dr. Jianwei Zhang, Universität Hamburg
Dr. -Ing. Jannik Fritsch, Honda Research Institute Europe
Dr. -Ing. Sven Wachsmuth, Universität Bielefeld

Gedruckt auf alterungsbeständigem Papier nach ISO 9706

Vision-Based Manipulative Gesture Recognition in a Human-Robot Interaction Scenario

Dissertation zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

der Technischen Fakultät der Universität Bielefeld
vorgelegt von

Zhe Li

23. April 2008

Acknowledgments

This work would not have been possible without strong support from many people. I would like to say thank you to those who have helped by giving me opportunities, advice, encouragement, and feedback along the way...

First of all, I would like to thank Prof. Gerhard Sagerer who not only gave me the chance to finish my PhD in the group of Applied Computer Science at Bielefeld University, but also supported me from all aspects in the past years and influenced me using his unique view on science.

Secondly, I deeply appreciated that I had the chance to conduct my work with two supervisors: Dr. Jannik Fritsch and Dr. Sven Wachsmuth. Dr. Jannik Fritsch brought me into the field of computer vision, equipped me with all fundamentals of how to do a research, and gave me valuable suggestions whenever i needed. Dr. Sven Wachsmuth became my second supervisor after Dr. Jannik Fritsch joined the Honda Reseach Institute Europe. In the last two years, he always instructed me and inspired my work with his deep understanding, wide-ranging knowledge and thoughts of computer vision and artificial intelligence. His words in our co-authored papers made them scientific and philosophic.

My thanks also go to the whole Applied Computer Science Group for forming a nice place to work, especially to Nils Hofemann, Lars Schillingmann, and Christian Peters for the pleasant cooperation in the project, to Joachim Schmidt, Marko Tscherepanow, Eduard Frese, and Fang Yuan for helping me at different stages of my work, and to all the colleagues who had participated in the experiments.

I'm very grateful to Prof. Ipke Wachsmuth and Prof. Jianwei Zhang that they agreed to join the defense committee despite their tight schedule.

Last but not least, I would like to thank my parents, who support me throughout my life, my wife Jing and our daughter Yunfei, who make the world much more colorful and meaningful for me.

Abstract

Nowadays, many people are expecting an age of personal robots just as what happened in the evolution of computers. With this background, the research on human-robot interaction receives a lot of attention in the robotics research community. In this dissertation, we focus on the vision-based recognition of human's manipulative gestures because the manipulation of objects draws the attention of the communication partner on the objects that are relevant for a performed task and furthermore the recognition of them serves the goal of a more pro-active behavior of the robot in passive, more observational situations.

Comparing to the interpretation of communicative gestures which can be recognized purely based on trajectory information, the understanding of manipulative gestures is more relying on the object contexts. Different to others, the approach we propose is called *object-oriented* w.r.t. three different aspects: it is object-centered in terms of trajectory features that are defined relative to an object, it uses object-specific models for action primitives, and it has an object-attention mechanism which is based on task models.

While most of the related work in gesture recognition assumes a fixed static camera view, such kind of constraints do not apply for mobile robot companions. After an analysis of the observational scenario, a 2-D approach was chosen by us. The manipulative primitive recognition scheme is able to generalize a primitive model, which has been learned from data items observed from a single camera view, to variant view points and different settings. We tackle the problem of compensating the view dependence of 2-D motion models on three different levels. Firstly, the trajectories are pre-segmented based on an object vicinity that depends on the camera tilt and object detections. Secondly, an interactive feature vector is designed to represent the relative movements between the human hand and the objects. Thirdly, a particle filter realized matching method adaptively finds out a scaling parameter which can fit the HMM-based models to different view angles.

To cope with different layers of intentions in the manipulative gestures, a unified graphical model with a two-layered recognition structure is proposed. The object-specific manipulative primitives on the lower level are coupled with task-specific Markovian models on the upper level. The combined bottom-up top-down processing loop in this structure realizes a dynamic attention mechanism by utilizing the task-level prediction of possible primitives to restrict the object types possibly detected as well as the action primitives possibly recognized. In this thesis, an online task-learning strategy based on prelearned object-specific manipulative primitives is also proposed. The task model can be initialized with few labeled data and updated incrementally when new unlabeled data becomes avaliable. The results of experiments in an office environment show the applicability of the approaches for vision-based manipulative gesture recognition put forward in this thesis.

Contents

List of Figures

List of Tables

1. Introduction

Recently, human-robot interaction is receiving more and more interest in the robotics as well as in the computer vision research community. From the robotics perspective, robots that cooperate with humans are an interesting application field that is expected to have a high future market potential. A couple of global and also mid-sized companies have come up with quite sophisticated robotic platforms that are designed for human-robot interaction. The ultimate goal is to place some robotic assistant or companion in the regular home environment of people, who would be able to communicate with the robot in a human-like fashion. As a consequence, the "hearing" as well as the "seeing" – as the most prominent and equally important modalities – are becoming major research issues.

The visual recognition of human actions is in the center of all these aspects and provides a bridge for a non-verbal as well as verbal communication between a human and the robot, which both are highly ambiguous. It enables the robot's anticipation of human actions leading to a pro-active robot behavior especially in passive, more observational situations. Furthermore, it draws attention to manipulated objects or places, embeds objects in functional as well as task contexts, and focuses on the spatio-temporal dynamics in the scene.

From the computer vision perspective, robot perception is more than an interesting application field. During the last decades, we can note a shift from solving isolated vision problems to modeling visual processing as an integral connected component in a cognitive system. This change in perspective pays tribute to important aspects of understanding dynamic visual scenes, such as attention, domain and task knowledge, spatio-temporal context as well as a functional view of object categorization.

A robot that is autonomously moving and acting in a human environment needs to understand and predict human behavior to a certain degree. While small automatic vacuum cleaners will mainly deal with collision avoidance for safety issues, larger movable robots, like the Bielefeld Robot Companion [44] need to respect human activities and situations beyond physical predictability leading to the recognition of human intentions. This starts by considering social spaces, detecting when a person does not want to be disturbed,

and ends in solving cooperative tasks with a human partner. The same accounts for human-robot communication starting with the problem to detect if and when a person communicates with the robot [63], via the interpretation of a communicative gesture [93] to the interpretation of the action context of an unspecific verbal statement [122; 3]. The reason for the increasing complexity in the interpretation of human motion patterns is the underlying factor of human intentions. The meaning of very similar human motions heavily depends on different layers of human intention. In this regard, Fleischman and Roy [30] argue that learning the meaning of verbs is much harder than learning nouns. They distinguish between two different kinds of ambiguities. (1) The vertical ambiguity refers to a possible causal chain of intentions, e.g. in order to get a cup, I need to find a cup, open the cupboard, and grab the handle. Thus, the same action "the hand moves to the handle of the cupboard" could be named on different levels of intention. (2) The horizontal ambiguity resambles that the high level interpretation could be ambiguous. For example, the same action as before could be interpreted as clean the cupboard instead of get a cup.

The different levels of intention have a different scope of interpretation in time and space. The physical prediction can be managed on a subsymbolic level considering the current trajectory of the human movement. Modeling social spaces needs at least some kind of representation of the human's mental state, while the recognition of actions like the opening of a cupboard needs to consider the relation of a human pose with regard to environmental objects and the changes of the object states over time.

The concept of different interpretation scopes directly fits Bobick's categorization of motion recognition: movement, activity, and action [8]. While movements can be characterized by reoccurring trajectories with a dedicated symbolic meaning, the interpretation of activities needs the extension of the scope in time in order to infer a higher level of intention. It represents larger-scale events, which typically include interactions with the environment and causal relationships. Actions involve a state change of the environment extending the scope into space.

So far we did not focus on the kind of body movement performed by a human perceived. A large amount of work is dedicated to whole body movements. An overview of several approaches is given by Gavrila [40]. Spatial as well as temporal contexts are considered by Intille and Bobick [51] in terms of multiperson actions and Fleischman et al. [29] in terms of places in an living environment. However, these approaches are mainly based on top-down views from surveillance cameras. In the robotics field most work is dedicated to gestures, i.e. intentional hand/arm movements that are mainly used for human-computer or human-robot interaction. A taxonomy of these is given by Pavlovic et al. [93]. They distinguish between manipulative and communicative gestures, on the one hand, and unintentional movements, on the other hand. Manipulative gestures are used to act on objects in the environment and, thereby, constitute actions, while communicative gestures are mainly characterized by a temporally structured activity. In this work, we are focusing on manipulative gestures. The manipulative gestures are hand actions that are defined by a non-deterministic sequence of object manipulations. They are often considered as the interaction between humans and their environment and being irrelevant to the communication between humans or humans and robots. From our point of view, the manipulative gesture also serves an important communicative function in human-robot

interaction. First, the manipulation of an object draws the attention of the communication partner on the objects that are relevant for a performed task. Secondly, it serves the goal of a more pro-active behavior of the robot in passive, more observational situations.

Much work has been done in the area of gesture-based human-robot interaction (HRI) because of humans' intensive use of their hands. These approaches mostly deal with symbolic, interactional, or referential gestures that have a communicative meaning on their own [85]. In terms of Bobick's taxonomy, they can be characterized as movements or, in more structured cases, activities. In this regard, object manipulations are more complex because the hand trajectory needs to be interpreted in relation to the manipulated object. In Fukuda's work [37], a cooking support robot is developed. It can recognize human manipulations of objects by sensing the movements of the markers on the objects and give recommendations by speech or gesture. Dropping these kinds of artificial constraints, the recognition problem is becoming notoriously difficult. Assuming that a hand is manipulating a spatially near object, it becomes hard to decide if the object is just passed by the hand or manipulated. Besides this segmentation ambiguity, there is a large spatio-temporal variability of how hand trajectories reach different object types and the appearance of a hand trajectory in a 2D image will also heavily vary according to the position of the object and the view-angle. Moreover, the mutual occlusion between the hand and the object causes even more difficulties for object detection and tracking. Therefore, the first point in this thesis is how to recognize the manipulative gestures despite huge variance given by the contiuous observation from a single camera. In order to have a better understanding of the manipulative gestures which could have different levels of intentions, more sophisticated schemes are needed that explicitly model contextual factors both in space and time. This is the second point in this thesis. The last point is how the robot obtains the model parameters by means of an incremental online learning process.

In the following chapters, the three points will be discussed in detail. Chapter 2 states the background of our work. It presents an introduction to the possible categorization and interpretation of gestures at first. Then, the system requirements according to our recognition scenario are analyzed. A discussion of various approaches of gesture recognition leads to a better understanding of our contributions in the area of manipulative gesture recognition, which is at the end of this chapter. Starting from bottom, Chapter 3 talks about the features used for the manipulative gesture recognition and the low-level image processing to extract them from the image sequences. Chapter 4 presents our models for manipulative primitives and how the object-specific manipulative primitives are spotted under spatio-temporal variability. In Chapter 5, the process not only goes a level higher, grouping the primitive sequences into manipulative tasks, but also has a feedback loop, refining the low-level processing using task level prediction. The recognition system is realized as a tightly coupled loop of bottom-up and top-down processing. Aiming for a natural robot learning, Chapter 6 puts forward a learning mechanism for the task model. Finally, Chapter 7 presents a summary of this thesis and gives an outlook to future work.

2. Background Statement

The great progress of modern science and technology, especially in electronics, robotics, artificial intelligence, and cognitive science, boosts the development of robots in the last decade. Today, when people mention robots, the impression is no longer dominated by the huge iron machines for industrial manufacturing or some cartoon characters. We can easily find some embodiments in our everyday life , such as the cute robot dog Aibo. Not only for entertainment, robots play different roles with different forms, like the automatic vacuum cleaner for housework, the robotic receptionist SAYA, etc. They observe the environment, listen to the parter and adapt their behavior like what humans do. Therefore, besides continuing in increasing the intelligence and capabilities of robots, the scientists also research on how to build up more efficient and natural interaction between human and them. Theoretically, the human-robot interaction (HRI) is a branch of human-machine interaction (HMI). Preceding HRI, HMI has been taken into consideration when the first automatic machine was created by human. At the very beginning, the comminication is carried out through switches, bulbs and punchcards. After the emergence of personal computers, the combination of mouse, keyboard and the graphic user interface (GUI) displayed on the monitor dominates the area for a long time. Nowadays, people look for more efficient and intuitive communications in HMI because it is expected that the intelligent robots do not only serve human as assistants, but also behave like partners or companions. It requires that the robot can understand the human like other humans do, even working under complex and variant environment, and interact with the non-expert user naturally.

With the development of intelligent machines, the human-machine interaction attracts more and more research interest and extends itself into an interdisciplinary field. From the view of computer scientists, the technique like speech recognition, image processing, and pattern recognition, etc. would enable nonintrusive communication between human and machine, which is similar to human-human interaction and less dependent on the apparatus for command input/output like keyboard. As a consequence, a lot of new communicative modes appeared in this direction, for example, dialog and gesture. With the help of these new modes, the user only need to say “follow me” or wave the hand to

him/herself to let a robot follow the user. The intelligent systems perceive the information from users by remote sensing, interpret it using pre-learned knowledge. Furthermore, a hybrid mode – the multi-model communication combines several HMI modes in a framework, which makes more complicate and smart systems possible, such as virtual environment [95], intelligent room [12], and all kinds of interactive robots [31].

In the robotic research society, the success of techniques for speech recognition and semantic understanding made the dialog a main communicative component used in many robotic systems [66]. As a nonverbal communicative mode, gesture conveys information in communication and in some cases it provide a simpler alternative for speech. For example, it is easier to point to an object than verbally describe its exact location. On this ground, some systems have been developed that the robots can receive the commands by observing the predefined characteristic hand gesture of a human. It is a huge step in the history of HRI. But they are not powerful enough to recognize the complex gestures in everyday life. Neither can they infer the human state by observing the changes in the environment caused by the hand movement.

Human Gesture has two basic functionalities: interaction and manipulation. The focus of this dissertation lies on the recognition of the manipulations. Humans make intensive use of their hands to interact with the environment in every life. Although the primary goal of such a manipulative gesture is the actual manipulation of the environment, other humans can observe the gesture to reason about the acting person "What is he/she doing". Nehaniv states "If the robot can recognize *what* humans are doing and *why* they are doing it, the robot may act appropriately" [85]. The recognition of manipulative gestures provides a good basis for the natural, pro-active, and non-intrusive interaction between humans and robots.

In order to understand the contribution of this dissertation, this chapter starts with the introduction to the general gestures including the taxonomy of gestures and the different layers of the interpretation of human hand movements. Then, the definition of manipulative gesture and how the recognition of it will benefit the HRI is put forward. As the research background, an application scenario of the manipulative gesture recognition on a robot companion platform is also proposed, which establishes the foundation of the system design and defines the scope of functionality of the expected system. Following the application proposal, the difficulties and keypoints to realize the vision-based manipulative gesture recognition are discussed in detail. After that, the state of art in relative areas is analyzed from different viewpoints, which leads to a better understanding of the contributions of this thesis, which are presented at the end of this chapter.

2.1 Comprehension of Gesture

What is gesture? In order to answer the question, we used the definition articulated by Matthew Turk [119]:

Definition 2.1 *Gestures are expressive, meaningful body motions-i.e., physical movements of the fingers, hands, arms, heads, face, or body with the intent to convery information or interact with the environment.*

It is said in the definition that the gesture includes the motions of different body parts. Hand gesture is only one certain kind in it though many researchers use the term "gesture" to indicate hand gesture because human use gesture for communicative purposes much more than other means. This definition also differentiates the gesture from posture, which is the static position or configuration of human body. More focusing on the dynamic motion aspect, Pavlovic gave a definition of hand gesture for the representation in computer [93]:

Definition 2.2 *Let* $\mathbf{h}(t) \in \mathcal{S}$ *be a vector that describes the pose of hands and/or arms and their spatial position within an environment at time* t *in the parameter space* $\mathcal{S}$*. A hand gesture is represented by a trajectory in the parameter space* $\mathcal{S}$ *over a suitably defined interval* $\mathcal{I}$*.*

From this viewpoint, he divided the gesture recognition into two main questions. The first one is the construction of the gestural model over the parameter set $\mathcal{S}$. The other one is how to define the gesture interval $\mathcal{I}$. Truly, the primary goal of gesture recognition is to create a system which can identify specific human gestures and use them to convey information. However, the hand gestures range from simple actions of pointing at objects to more complex ones that express our feelings or perform a task. In order to model the hand gestures properly, it is necessary to have a close look at the different types of hand gestures and their associated properties. Moreover, the understanding of a hand gesture has huge variance under different interpretation scopes, which heavily depends on human intentions. In the following of this section, the taxonomy of hand gesture and the different interpretation level of hand motions will be discussed in detail.

2.1.1 Taxonomy of Hand Gesture

Because of the tremendous use of gesture in human-human interaction, it has been investigated by linguistists, psychologists, and sociologists for a long time. Several taxonomies have been suggested in the literature. Cadoz uses functional roles to group gestures into three types by their functionalities [119]:

Semiotic – to communicate meaningful information

Ergotic – to manipulate the environment

Epistemic – to discover the environment through tactile experience

From the point of view that gesture is intimately related to speech, Kendon described a "gesture continuum" [57], shown in Figure 2.1. It defines five different kinds of gestures: *Gesticulation* – spontaneous movements of the hands and arms that accompany speech; *Language-like gestures* – spontaneous movements of the hands and arms that accompany speech; *Pantomimes* – gesticulation that is integrated into a spoken utterance, replacing a particular spoken word or phrase; *Emblems* – familiar gestures such as "V for victory"; *Sign languages* – Linguistic systems, such as American Sign Language, which are well defined. As the list progresses (from left to right in the figure), the association with

Figure 2.1: Kendon's gesture continuum

speech declines, language properties increase, spontaneity decreases and social regulation increases.

Although the attempt to understand gestures can be traced back to the work of Bulwer on chironomia in 1644 [16], only in the last several years has there been an increased interest in trying to introduce gesture into the field of Human-computer interaction (HCI). Pavlovic classifies all hand/arm movements into two major classes: gestures and unintentional movements [93]. Gestures are further divided into two modalities: communicative and manipulative. Manipulative gestures are the ones used to act on object in an environment. Communicative gestures have an inherent communicative purpose and are usually accompanied by speech. Comparing the categorization by Kendon and Pavlovic, it is found that Kendon's understanding of gesture is corresponding to the subgroup of "communicative" in Pavlovic's taxonomy (Figure 2.2 shows a complete structure).

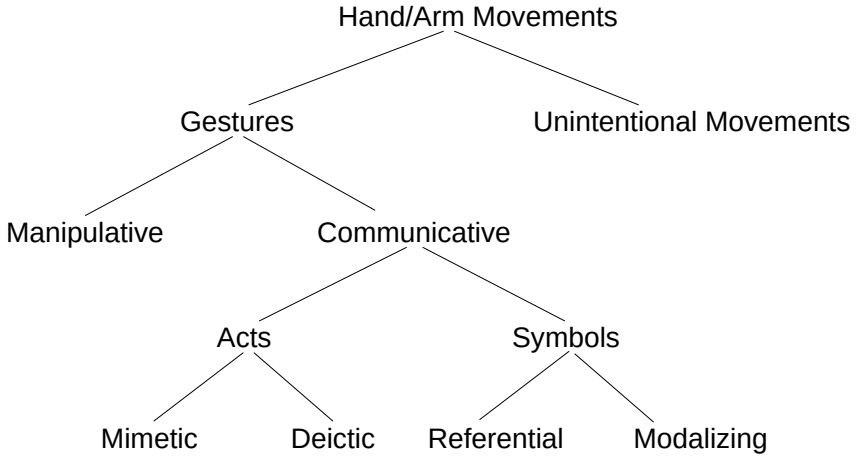


Figure 2.2: Taxonomy of gesture for HCI from [93]

In more recent studies, Nehaniv put forward a coding scheme to identify people's gestures when asked to explain a home task to a robot [85]. He argued that in order to infer the intent of a human interaction partner, it is useful to classify the gestures into major types. The five classes in his categorization are as following.

Irrelevant and Manipulative Gestures: these gestures do not have a primary communicative or interactive function (in practice, this class is split). The former subclass is not relevant for most HRI purposes, but exclusion from consideration following recognition is desirable. To the contrast, manipulative gestures change the environment or human's relation to it.

Side Effect of Expressive Behavior: these are gestures that occur as side-effects of people's communicative behavior. They can be motion with hands, arms, face, etc but without specific interactive, communicative, symbolic or referential roles.

Symbolic Gestures: these are gestures that follow a conventionalized signal. Their recognition is highly dependent on the context (both current task and cultural milieu).

Interactional Gestures: this category classifies gestures used to regulate interaction with a partner. Thus they can be used to initiate, maintain, invite, synchronize, organize, regulate, or terminate an interaction behavior between agents.

Referencing/Pointing gestures (deixis): the gestures that fall into this category are gestures used to indicate objects or loci of interest.

Nehaniv also said there are some gestures which have aspects from more than one class, for example, handing over an object is both manipulative and interactional gesture. Clark expressed a similar argument in his work [22].

Most of the previous gesture-based interaction systems use constrained and task-specific gestural primitive repertoire. Nehaniv is the first one trying to identify gestural classes for HRI [84]. This work poses the question – when a gesture is observed, in which scope shall a robot interpret it?

2.1.2 Interpretation of Hand Motion

Human has an inherent ability to understand dynamic motions. Gergely showed that even one-year-old infants can infer intentions before the whole activity is completed[41]. But this capability is still ambitious for the current intelligent systems. For example, when seeing a basketball player dribbling a ball forward in a match, our first impression is she/he is attacking. The up-to-date vision-based recognition systems could interpret the hand movement as "is moving" or a repeat of moving up and down, but not dribbling, or the same as the human's without the preknowledge about basketball and the concept of "match". When the research question shifted from "how are things moving?" to "what is happening?" [7], Bobick gave a hierarchy to interpret the motions of human body [8].

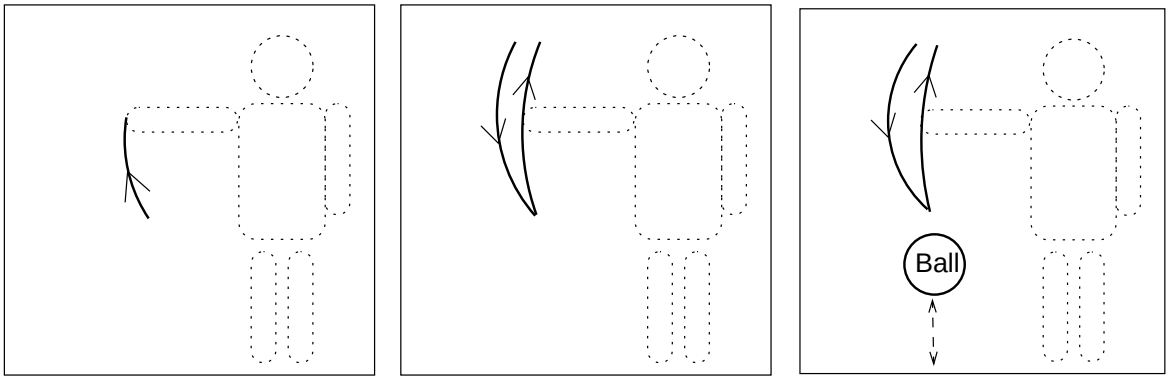
Movement: they are the most atomic primitives, requiring no contextual or sequence knowledge to be recognized; movement is often addressed using either view-invariant or view-specific geometric techniques.

Activity: refers to sequences of movements or states, where the only real knowledge required is the statistics of the sequence.

Action: are larger-scale events, which typically include interaction with the environment and causal relationships.

Figure 2.3 drawn by Fritsch [33] shows a symbolic sketch of a human performing three different hand motions relating to the three categories. Only the information represented

by solid lines is available to the algorithms for performing motion understanding, indicating the human by a dotted line is done to facilitate an intuitive example. The hand going up in Figure 2.3(a) is a movement, it can be represented by the motion of the hand only. Its execution is consistent and easily characterized by a definite space. The gesture in Figure 2.3(b) is an activity, as it consists of a sequence of up and down hand movements, with arbitrary delays inbetween. Recognition of such a motion requires knowledge about both the appearance of each constituent movement and the statistical properties of the temporal sequence. Like movements, activities do not refer to elements external to the actor performing them. Finally, Figure 2.3(c) depicts the same hand motions, but now there is an additional symbolic information "ball" present. To recognize this motion as action "dribbling", the trajectory data has to be linked to the object "ball" in its vicinity. Without such a context, the interpretation of motion is ambiguous. Therefore, Bobick describes actions as being "at the boundary of where perception meets cognition."



(a) Movement (b) Activity (c) Action

Figure 2.3: Images demonstrate Bobick's categorization of gestures based on the knowledge necessary for recognition [33]

Nehaniv also emphasizes the importance of the context in understanding hand motions [85]. He said it should help the gesture recognition when objects, humans and other animated agents in the environment are identified and tracked, previous and current interaction patterns are remembered to predict the likely current and next behavior of the particular person, and the scenario and situational context are known (e.g. knowing whether a gesture occurs at a tea party or during a card game). Hofemann deliberated the gesture context from three aspects [47]: symbolic, situational and social, shown in Figure 2.4. The symbolic context represents the spatial relativity between the gesture and individual subjects, like objects. The situational context are more focusing on the causality existing in the gesture sequences and the scenario. The social context indicates the prejudgments which affect the comprehend of gesture like common understandings in a society, personal experience, etc. Thus, for a robot who wants to understand human gestures, it is necessary to associate the motions with appropriate contexts.

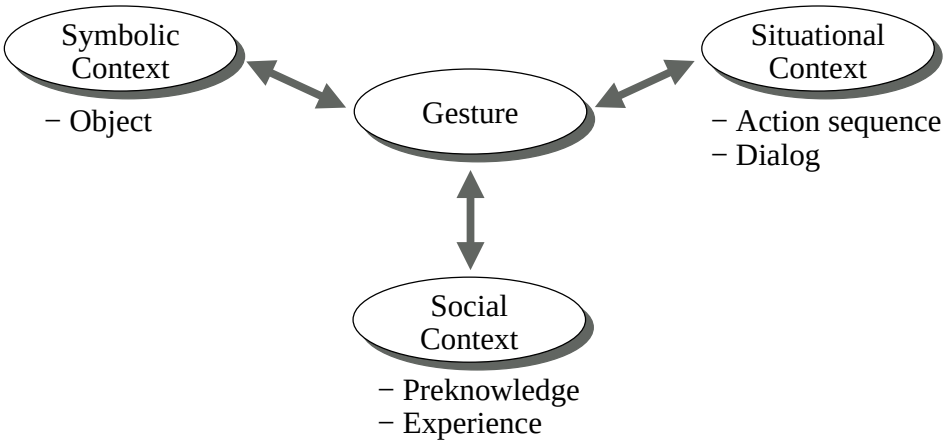


Figure 2.4: Interpretation of gesture context [47]

2.2 Manipulative Gestures

Some researchers define the gesture based on its communicative functionality. They argue that manipulations are not gestures because they are not utilized for communication. Billinghurst and Hulteen said "A gesture is a motion of the body that contains information. Waving goodbye is a gesture. Pressing a key on a keyboard is not a gesture because the motion of a finger on it's way to hitting a key is neither observed nor significant. All the matter is which key was pressed" [62]. We agree with this argument if and only if what the system need to know is the key index. However, for human or a household robot which observe the state of human, manipulation contains much more information. For example, a pressing could be carried out lovingly or in anger. In general, manipulations may not executed for communicative purposes. In Pavlovic's taxonomy of gesture [93], the manipulative and communicative gesture also stand as two parallel categories. But they do contains the information which need interpretation in HRI. What is worth mentioning is the overlap between the manipulative and communicative gestures. In the work of Clark [22], a deep analysis of an communicative manipulation – "placing" is presented. A simple example is that a purchase can be confirmed by placing the money on the counter.

According to Definition 2.1, manipulation is absolutely a kind of gesture. A concrete definition is given by Quek [100]:

Definition 2.3 *a manipulative gesture is one whose intended purpose is to control some entity by applying a tight relationship between the actual movements of the gesturing hand/arm with the entity being manipulated.*

As mentioned before, it can be communicative or ergotic. Our work is focusing on the latter, the manipulations not intended for communication, because we are focusing on the case that the robots are in an passive, more observational state. By recognizing the manipulative gestures, the robot could achieve attention to specific objects in the scene, monitor the user state, analyze the task, and infer the intention of user, etc. For example, developed the cooking support robot developed by Fukuda et al. [37], can recognize human

manipulations of objects by sensing the movements of the markers on the objects and give recommendations by speech or gesture.

The manipulative gestures have a variety of forms, from basic actions like touching, grasping to structured and more complicated cases like making a cocktail, from the real manipulations in everyday life to the mouse actions on a computer desktop or the motions in a virtual environment. They are different from each other in the temporal, spatial aspects, the complexity, the manipulated objects, the contexts and the environments, etc. In order to understand the work in this dissertation, the following subsections will give a close view of the manipulative gestures in our attention, present the proposed recognition scenario, and discuss the difficulties in a vision-based approach.

2.2.1 Discussion on Manipulation Recognition

In robotics, the word "manipulation" often denotes the movement of robot hand or arm accomplishing a task, for example, a robot takes up a cup from a table. The main reason is that the research on manipulation in robotics dedicates to the motion control, with which a robot can help people finishing different kinds of tasks. Tightly related to robotics, in HRI "manipulation" is used to indicate the motions of human hand. Robots can estimate the status of the user by recognizing their manipulative actions. The different concepts cause two different research questions on robots: how to perform an action and how to recognize an action performed by human. The motion of a robot results from the control signals sent to its motors. The control signals are either hardcoded or generated by a learning process. The learning process could be a direct mapping of the motion between the teacher and the student when both sides have the same embodiments and the signals from the attached sensors on the teacher can be transferred to the motors of the student, e.g. use cyberglove to control a robot hand [58]. Another learning mode is the so-called "learning by observation" (LBO) [26; 117]. It is much harder than the former mode because the signal transformation from external appearance to precise internal control is difficult. But it is gaining more and more research interest because of the intuitive teaching style. The learning of how to recognize human actions has similar approaches with the latter. But the difference is that in the HRI the goal is to recognize an action with a certain generality, not to repeat an action with high accuracy.

Manipulative gestures differ from the gestures used in communication in several ways. Firstly, the performer has tactile sensation in the hand and force feedback from the object being manipulated. But when manipulations are observed by remote sensing techniques such as a camera, the force feedback to human fingers is not available to the receiver. Therefore, in a vision-based approach for manipulative gesture recognition, the trajectories of the human hand are more salient features than the pressure between the hand and the object. Secondly, the hand motions during manipulations are affected by the objects and their positions. For example, when taking a cup filled with water, the hand will move slowly, when taking an empty cup, the hand will move faster. That means when we interpret the hand motions, the related objects must be taken into consideration, too. Thirdly, there is no guarantee that the performing hand is always completely visible to the observer. The occlusion between human hands and objects make the tracking of individual fingers infeasible and the appearance of the performing hand in images is deforming during

manipulation. In this sense, the hand is represented by a blob in the images instead of concrete 3-D models. The visual recognition of manipulative gesture is then to classify the dynamic hand trajectories like "hand approaches a cup" rather than recognize the nearly static manipulative state like "a cup is held in hand".

Manipulative gestures are also distinguished from other gestures by the fact that their logic and meaning are particularly clear. The sequential manipulation of "take a pencil" and "move it to a notebook" should not only be recognized as two independent manipulative actions but also as an entity with the underlying human intention "to write". Some researchers proposed theories of semantics and inference of action [18]. Being primarily focused on computer vision our goal is to stay as connected to the visual signal as possible, where the particular semantic elements for example "take a cup" have direct visual correlates.

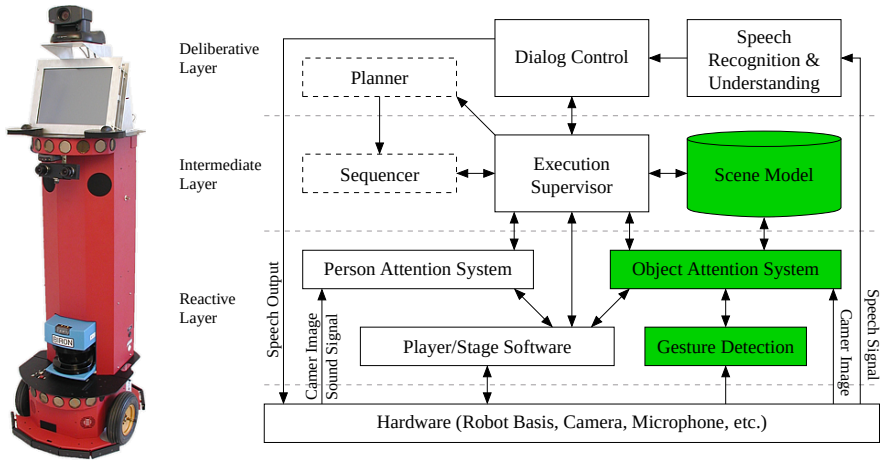
2.2.2 An Indoor Scenario for Observation

Fukuda proposed a kitchen scenario for a cooking support robot [37]. All the objects are attached with an IC tag. So the manipulation can be detected by sensing the movements of those markers. In Moore's work, a survilliant environment is set up to monitor the indoor manipulations [80]. The cameras are mounted on the roof the rooms. Different to those systems, our platform for the manipulative gesture recognition is a household robot. These kind of robots are designed for serving humans as a companion in their daily life. They are expected to act in an unstructured environment, such as an office or a private home and communicate with nonexpert users in natural and intuitive ways.

In our work, the focus is on the object manipulations which happen in everyday life. The recognition of such manipulations leads to a more pro-active behavior of the robot in passive, observational situations. In order to achieve non-intrusive recognition, we choose using the camera of the robot to avoid special markers or sensors on human or objects. In domestic environments many actions are performed on a table top (e.g. preparing a meal, decorating a table, performing typical office work, watering flowers). A common scenario that well motivated is that the human sits at a table with normal height as a desk in an office or a dinner table at home and performs daily housework or normal desk work on it. It is assumed that a mobile robot moves to a place around the table where it is able to observe the sequence of actions in focus and the field of view of robot camera is broad enough that it need not to adjust its position or shift the camera to follow the movement of human hand during the performance.

Because the robot moves around in the room and the user does not always sit at the same position, it will not be feasible for the robot to go back to the same position and observe the user from the same viewpoint every time. So the observation need to be view-variant and from different distance.

The robot which our approach is intended for is the Bielefeld Robot Companion (BIRON) (see Figure 2.5(a)), which is described in [44], [128] and [35] in detail. Its hardware platform is a Pioneer PeopleBot from ActivMedia. It has three PCs for controlling the motors and on-board sensors, image processing and speech processing and dialog separately. A pan-tilt color camera (Sony EVID31) is mounted on top of the robot at a height of 141



(a) BIRON (b) Architecture

Figure 2.5: The Bielefeld Robot Companion (BIRON) and its system architecture

cm for acquiring images. It is the camera used by us for image perception. The system architecture is shown in Figure 2.5(b) [61]. The modules which the manipulative gesture recognition concerns are indicated by color. In the figure, it can be found that the gesture recognition is not a isolated module which performs a pure bottom-up processing. The scene model and object attention system provide top-down information and construct the gesture context. The combined bottom-up and top-down mechanism will be deliberated in Chapter 5.

2.2.3 Corresponding Problems in Vision-Based Approach

Visual interpretation of hand/arm movements has a tremendous advantage over other techniques that require the use of mechanical transducers: it is non-intrusive. Figure 2.6 shows the block diagram of vision-based gesture recognition systems. First of all, the subjects of interest from the input images must be detected, for example, the human hands and the objects. Then their states, such as positions and velocities, are tracked over time and fed into the recognition module. Recognizing gestures from these data is a pattern recognition task which typically involves transforming the input into an appropriate representation (feature space) and then classifying it based on a database of pre-learned gesture models. This diagram is simple. Nevertheless, the implementation carries a burden of different challenges.

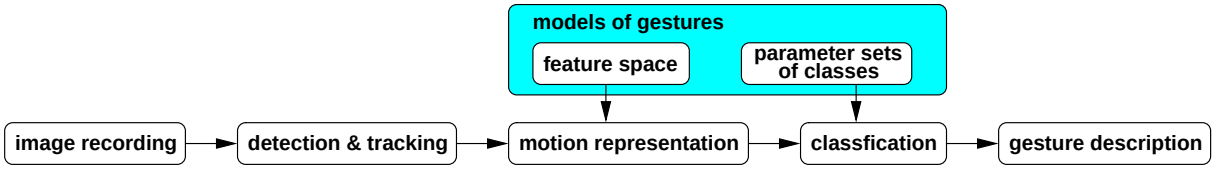


Figure 2.6: Block diagram of a vision-based gesture recognition system

The gesture modeling and recognition are essential to the interpretation of human gestures. However, several imaging issues affect them from the very beginning: what kind of camera

is appropriate for the manipulative gesture recognition purpose, mono-camera, stereo-camera or multi-camera systems? does the camera stay at a fixed position or will it move during recording? and from which view-angle are the performances observed? As described above, a pan-tilt mono-camera is used on our robot platform. Compared to other commercial products, this camera satisfies the system requirements and is economical. In the proposed scenario, a fixed camera position is assumed throughout one observation. But the *view-angle* varies over different observations. In most of previous works, the problem of view-angle was ignored or a fixed viewpoint was assumed. It is challenging to take the view-angles into consideration in gesture recognition when the gestures are observed by a mono-camera because the variant projections of the same motion on 2-D images with regard to the different view-angles result in a huge variance of the trajectories. In addition, the *distance* between the camera and the human also affect image-based measurements. Due to the pin-hole camera model, more far away the human is, more shorter the same motion of the human will appear in the images. That means this motion will be perceived by fewer pixels on the image. The effect of observing a motion far away is equal to down-sampling the motion trajectory which is perceived in the near.

The next step after recording the images is to find the regions of interest from them. The region of interest could be a moving area, an image patch with skin color, a concrete object, a complete human body etc., which is dependent on the representation of targeted gestures. In our case, detecting human hands and objects in the scene are the main focus. Although many methods and algorithms have been developed for these purposes during last decades, their applications are still limited in well-defined environment and facing different challenges, for example, detecting human hands using color cue under a various illumination conditions, recognizing the objects belonging to the same type but with different appearances, etc. Moreover, human can easily recognize a known subject even if it is partially occluded. But the occlusion is still a huge obstacle against robust object detections in computer vision systems.

Motion representation and classification is the central module of a gesture recognition system. It contains the predetermined models and the model parameter spaces for different classes. In order to classify the observed gestures into different clusters correctly, the parameters in the gesture models should be representative in the proposed application scenario and their space ought to be well-separated from each other. The parameters – position and speed of the hand blob in the image coordinates are the intuitive representations of the hand motion. It can be used for a fixed camera view. If there are possible translation and rotation of the images, the relative position and the direction of the hand velocity are more appropriate for the recognition task. Of course, there are also a lot of other representations developed for different scenarios. For example, Bobick and Davis introduced the motion-energy images (MEI) and motion history images (MHI) [10], which indicate the duration of motion at certain position in the images. Combining the MEI templates from different view-angles, a view-invariant gesture recognition is achieved. Because of the various object positions and view-angles in our scenario, the choice of features for manipulative gestures must take the translation, rotation and view-invariance of the motions into consideration.

Before matching the observed feature sequence to the models, the step *segmentation* is performed when the input to the classification module is a long-term observation. On one

side, the long-term observation could contain different meaningful parts. It is no sense to match it to individual models. On the other side, eliminating the trajectories which are meaningless motions could decrease the system's processing load. It is awkward to insert or define special states indicating the start and end of a gesture into continuous performance of human. Due to the ambiguity in the segmentation, instead of solving the problem before matching, some researchers try to couple the segmentation to the matching by searching the optimal start and end along the trajectory using dynamic programming or other methods [1; 65]. This is also called gesture *spotting*.

Even if the difficulty in segmentation is solved, there are several well-known challenges when matching the segmented observations to gesture models – *spatial variance*, *temporal variance*, and *subgestures*. The spatial variance has two aspects. Firstly, repeated performances of the same activity by the same human vary even when all other factors are kept unchanged. Secondly, similar activities are performed by different individuals in different ways. The temporal variance means that similar activities can be of different temporal durations and also different activities may have significantly different temporal durations. Subgestures are those gestures which are a part of another gesture, but also carry their own meanings as individual entities. Because of their duplicated appearance in different gestures, they bring much ambiguousness into the segmentation and classification.

From the observations to the descriptions understandable by humans, this gap is also known as the *symbol grounding problem*. In a long-term observation, specially a sequential object manipulation, human can have different interpretations of the images according to different internal intentions. Therefore, the research question are on which level the symbol grounding should be established when designing a system to recognize complex hand actions and further more, if the symbols grounding is built up, how to achieve a higher level comprehension of an observation from a sequence of symbols by modeling their semantic, causal, or probabilistic relationships.

As mentioned in Section 2.1.2, the context plays an important role in the gesture interpretation. Especially, understanding manipulative gestures only based on the hand motion without object information is infeasible. But on the other end of the spectrum, there is also an *attention* problem if there are many objects in the scene. An appropriate attention on certain objects could lead to more precise recognition by getting rid of the noise from unnecessary processing.

2.3 State of Art

In Section 2.2.3, the main difficulties for developing a vision-based manipulative gesture recognition system are discussed. Before describing the contribution of this dissertation on this topic, the current research state will be reviewed from several aspects.

2.3.1 2-D and 3-D Gesture Recognition

Based on whether modeling the motion of gestures on the image plane or in the real world the research work is divided into two approaches: 2-D and 3-D based. The former can be thought of as an approach which bypasses a pose recovery step and describe the gestures in terms of simple low-level, 2-D features from a region of interest. Polana and Nelson

refer to it as "getting your man without finding his body parts." [98] This approach has been used in sign language recognition [115], gesture based interface [123], where only the hand blob were tracked over time. A priori knowledge of how the human body/hand appears in 2-D images could benefit the detection and tracking. This approach is named 2-D approach with explicit shape models. The challenges for 2-D systems are the possible occlusion to the region of interest and the different projections of the same motion on the image plane due to variant view-angles of the observing camera.

An intuitive solution to the view-angle problem is to represent the motions in 3-D space. However, the general problem of 3-D motion recovery from 2-D images is quite difficult because there is no precise depth information in 2-D images. Therefore, the images from stereo- or multi-cameras are used. Combining the projective functions of different view-angles, the 3-D position of a point in 2-D images can be computed. Some researchers used 3-D information to recognize the human motions [133].

It is fair to say that the results of vision-based 3-D tracking without markers are still limited at this point. To achieve better results, multicamera system with large size is needed. But this requirement is too restrictive for most of the robotic platforms. One approach is given by Schmidt et al. to track an elaborated 3-D human body model in monocamera images by fusing multiple cues (edge, ridge color) [112]. This approach suffers from the initialization problem, which is currently unsolved.

Attempting to recognize actions in 2-D images from different view-angles, Rao introduced the view-invariant features – the so called *dynamic instants* [102]. They are the dramatic changes of the spatio-temporal curvature of a 2-D trajectory. He argued that the same action should have the same number of instants. But the trajectories of object manipulations can have very different dynamic appearances because of the different positions of the objects and the unpredictable movements which are far away from the objects. Therefore, the dynamic instants proposed by Rao are not sufficient to describe the anticipated manipulative actions.

2-D approaches are effective for the applications where human performs constrained movement and single viewpoint is assumed. A 3-D approach seems more appropriate for recognizing complex human movements because it provides a more accurate, meaningful representation of physical space which allows a better handling of occlusion and leads to better discrimination between similar motions.

2.3.2 State-Based and Trajectory-Based Approaches

In order to narrow the gap between image perception and scene understanding, the image contents are presented by symbols understandable by humans, such as the image of a cup mapped to the label "cup", the motion of a human hand approaching a cup interpreted as "take cup". After having labeled all items, every subsequent task is solved by manipulating the labels. In some works, the mappings from real objects onto symbols were done manually, leaving only the symbol manipulation to the machine. For example, Yamamoto developed a system to recognize different Japanese temaes – a set of rules for serving tea. The meanings of action segments are given by human. Then, different sequences are classified by using Stochastic Context Free Grammar (SCFG) [132]. As it turned out, this symbol grounding problem is not solved.

In computer vision society, one of the tasks is to develop the algorithms which can extract interpretable symbols from the images automatically. In the case of gesture recognition, the motions will always be classified into a set of classes, which are indicated by symbols. Generally, the approaches for the scene understanding are divided into two categories according to their different symbol grounding solutions.

The first category is *state-based*[1] approaches. The states can indicate both static scenes like "hand stop" and dynamic processes like "approach a cup". They are generated by detecting one or more key states, without using explicit motion models. In the assembly construction scenario [36], the states used are the presence of the elementary objects and the assemblies. The assembly recognition is achieved by detecting the changes of the states – the appearing and disappearing of the parts[2] in the scene. Jo used a Finite State Machine (FSM) for modeling possible state transitions in the manipulative gesture [55]. In his approach, not only the states such as "hold", "release" and "rest" but also the transition conditions between them are predefined. Instead of searching the states from the images directly, the system does the state transition when corresponding conditions are satisfied.

The other category is *trajectory-based* approaches. Different to state-based methods, the symbols here are extracted by matching the trajectories to predetermined motion models. The "trajectory" doesn't mean that it is only a sequential representation of the positions of points on a 2-D coordinate. It can be the tracking of any features. In McKenna's visually mediated interaction project, trajectory templates are built up for teleconferencing camera controlling e.g. pointing left and pointing right based on the three features – the motion area, the centroid displacement and the elongation of the motion [76]. The matching of these templates are done in the framework of CONDENSATION [6]. Lee used a set of gesture commands to browse microsoft powerpoint [65]. The motion feature used is the vector quantized direction of hand motion. The meaningful trajectories are spotted by a threshold model based on hidden Markov model (HMM). In order to robustly recognize the trajectories under the performance variance (see the discussion in Section 2.2.3), the symbols in trajectory-based approaches normally denote only well-formed, short processes, which are named *primitives*, or *atomic actions*. It is seldom that complex and long-term actions are coded by trajectory models.

For a long-term action which consists of several primitives, the hierarchical model is a common choice. The symbols are assigned through the low-level image processing and the sequences of the symbols are modeled on higher level by grammar or probabilistic models. In Chan's work, a simple feature vector is used for modeling the interaction primitive, e.g. *approach*. The transition of the semantic primitives are modeled by HMM [18]. Because of the early symbolic abstraction of trajectory information, this method can only be applied in a restricted scenario. Bui introduced the abstract hidden Markov model (AHMM) to recognize the office work which happens in several rooms on a floor, e.g. "go to the service room to take the printed document back" [15]. The trajectories in a room and the visiting of different rooms are all modeled by HMM on different layers.

[1] It is also named *symbolic-based* approaches by some researches. Here, the concept of symbol is different to that in symbol grounding because it represents only static states.

[2] the term "part" denote either an elementary object or an assembly made from elementary objects

In sign-language recognition, hand trajectories are the dominant features. For emblem gestures like "v" for victory, the static posture are more prominent. In some unrestricted scenarios, like visual surveillance, video conference, neither trajectories nor some static states are sufficient for representing the features. Therefore, there is a tendency to combine the symbolic and trajectory-based approaches for the symbol grounding problem. In the recent research on video-based event recognition, Hongeng et al. [49] proposed that an activity is considered to be composed of action threads, each thread being executed by a single actor. A single thread action is represented by a stochastic automation of event states. The states are recognized from the characteristics of the trajectory and the shape of moving blob of the actor using Bayesian methods. Similar to Hongeng, Robertson used Bayes nets to fuse the samples from the the non-parametric training databases of trajectories and local motion descriptors [105]. To recognize manipulative gesture, Moore put forward the concept of objectspaces, which integrate sensory trajectory data and symbolic object data together [80].

2.3.3 Gesture Recognition with Context

As described in Section 2.1.2, the context plays an important role in the interpretation of hand motions within different scopes. For example, the extracted symbols from the images could augment the meaning of the hand motion. In the attempts to add commentary to tennis videos automatically, Robertson used the local motion descriptor and also the location together with motion trajectory samples to infer the possible actions [105]. In Moore's work, a camera mounted on the ceiling observes a human interacting with different objects in a home or office scenario. He used the objectspace to combine both types of information, sensory trajectory data and symbolic object data, in a structured framework [80]. Certain image processing steps are carried out to obtain image-based, object-based, and action-based evidences for objects and actions, which are integrated using Bayesian networks. Action primitives are recognized from hand trajectories using HMMs that are trained offline on different activities related to the known objects. While this approach centers a context area around detected objects, hand-centered methods define context areas relative to a hand trajectory. Fritsch et al. [34] put forward such an approach. In this case, the trajectory information is augmented in each time step by contextual objects that are searched on-line using the context area bound to the moving hand. This method is also implemented in a pointing recognition system [46].

On the situational level, the context could come from many aspects, e.g. the actions happened before, the parallel activities from other agents and the scenario for the actions etc. Focusing on the temporal structure of the action performance, the state-space models such as FSM, HMM and the extensions of HMM and grammar models are often used [48; 25; 74; 86; 99; 78]. Implicit in the models is the assumption of strictly sequential sub-actions, which are adequate for the mapping into primitives. Bobick developed a PNF (past-now-future) constraint network to model the temporal structure of actions and subactions with overlappings [96]. An application is presented by Yu et al. [135]. They argue that the eyes guide the hand in almost every action or object manipulation. The gaze could be a context of the action to eliminate the effect of meaningless observations. In their work, the eye motion is measured by a head-mounted eye tracker and used for the segmentation of hand trajectories and the detection of objects. HMMs are used for

action recognition which is purely based on trajectory information. The object and action information are integrated on a symbolic level using action scripts.

The scenario as context can also be used to classify the ambiguous actions, e.g. "take a cup" in the office is a hint for drinking, but the same action in a bathroom could be used to infer "teeth brushing". Once the scenario is known for a recognition system, the possible actions could be predicted and the attention to the key subject could also be achieved. In the modeling, it is normally taken as top-down (or goal-directed) approaches. Oliver proposed a system framework which combines top-down with bottom-up information for visual surveillance [91]. But in that paper, no clear solutions for the top-down prediction have been provided. Khadhouri used this mechanism on the robot's attention problem [59]. It is proved by his experiment that more confidence can be achieved on the correct action with this mechanism.

Moreover, the social context such as the personal experience, the common sense existing in the society, and the culture affects our everyday judgment. But on the state of art, the models on this aspect are seldom in that area of vision-based action recognition.

2.3.4 Task Learning

Learning tasks from human demonstration is crucial because it is important for human-centered robots to adapt their behavior according to the different situations and user personalities. In the robotic society, there is always the expectation that the ordinary people, not professionals, teach the robot tasks. The easiest and desirable way of teaching is to just demonstrate the behavior so that the robot automatically learns it from observation and builds an abstract representation of the task. The framework is called "learning from observation". There are two approaches dedicated to different aspects: one is the *imitation learning* in which the demonstrated trajectories are directly transformed to robot motion, the other is *background knowledge based or deductive systems* which designs a set of task-dependent primitives and tries to recognize the subject task as a sequence of symbolic primitives. Because our focus does not lie on the precise repetition of the user motions but the interpretation of human actions, only the latter approach will be discussed.

In the smart homes project, the inhabitant actions of the user are predicted according to task models, which are Markov models that have been generated from an unsupervised clustering of the data recorded in 1250 days [103]. For a household robot, it may be feasible to group similar activities after observing a certain amount of tasks for many times. But more naturally, users will expect that the robot is able to learn some tasks from only a few demonstrations. Aiming to teach a robot how to perform everyday manipulation tasks, Ogawara proposed the task model which is a sequence of essential interactions [87]. In his definition, an essential interaction happens between two objects which consists of the grasped object, the target object and the relative motion between them and extracted by the robot from multiple demonstrations of one task. The trajectories are clustered and represented by HMM [88]. More focusing on the causality among the subtasks, Pardowitz developed a task precedence graph (TPG) for task modeling [92]. It is a directed graph with a set of subtasks as basic elements and the directed connection between two subtask indicates a task execution rule which must be complied with. The TPG could be initialized

as a most restrictive model with one instance and generalized by dropping unessential precedence relations when new demonstrations are available.

The approaches above, expecially Ogawara and Pardowitz, depend on a robust detection of the elementary actions in the tasks. The robustness in their systems is realized by using different sensors simultaneously, such as dataglove, the stereo vision system with multi-cameras. However, a mobile robot prohibits the use of attaching sensors and large-scale camera systems.

In the visual surveillance scenario, much work in the recognition of human activities uses stochastic models to cope with the uncertainty in the detection of primitives. HMM and its extensions have become very popular in this area [25; 74; 28; 135; 13; 86; 130]. Other than using probabilistic models, activities with predefined context or inherent semantics are also represented by event-based predicate logic [27], deterministic action grammars [11], stochastic context-free grammars [54] etc.

As state of art, a full theory and specification of all human behaviors is still beyond the capabilities of current knowledge representation and reasoning systems. The task modeling and learning must be done in a reasonably constrained environment.

2.4 Contributions

In this thesis, we are trying to provide a solution for a mobile robot to recognize the human's manipulative gestures from different single camera views. One important feature of manipulative gestures is that they are interactions between human and objects. Comparing to the interpretation of communicative gestures which can be recognized purely based on trajectory information, the understanding of manipulative gestures is more relying on the object contexts. Different to the hand-centered approach put forward by Fritsch et al. [34], the approach we propose is called object-oriented. This is similar to the object represention scheme from Moore et al. [80]. What goes beyond their framework is the consideration of different perspective views as well as the spotting of meaningful parts in longer hand trajectories.

While most of the related work in gesture recognition assumes a fixed static camera view, such kind of constraints do not apply for mobile robot companions. Instead of using 3-D approaches, we chose 2-D representation for the manipulative gestures. A manipulative primitive recognition scheme is proposed that is able to generalize a primitive model, which has been learned from data items observed from a single camera view, to variant view points and different settings. We tackle the problem of compensating the view dependence of 2-D motion models on three different levels. Firstly, we pre-segment the trajectories based on an object vicinity that depends on the camera tilt and object detections. Secondly, an interactive feature vector is designed that represents the relative movements between the human hand and the objects. Thirdly, we propose an adaptive HMM-based matching process that is based on a particle filter and includes a dynamically adjusted scaling parameter that models the systematic error of the view dependency.

To infer higher level intentions in the manipulative gestures, we propose a unified graphical model with a two-layered recognition structure. On the lower layer, the object-specific

manipulative primitives are represented as HMMs which are coupled with task-specific Markovian models on the upper level. In this structure, we have a combined bottom-up top-down processing loop. Thereby, a dynamic attention mechanism is realized that utilizes the task-level prediction of possible primitives in order to restrict the object types possibly detected as well as the action primitives possibly recognized.

In this thesis, we also propose an online task-learning strategy based on prelearned object-specific manipulative primitives. The task model can be initialized with few labeled data and updated incrementally when new unlabeled data becomes avaliable. Moreover, with the possibility to reject, the system is able to detect the unseen tasks during learning process and build up new task models. These two capabilities – to reject unmodeled sequences and to learn from unlabeled data – are essential for a human-like interaction style with a robot, which does not separate between a learning mode and a recognition mode. Learning of new task models can take place during normal interaction. If the robot does not understand a specific action sequence the human interaction partner can instantly react on it by repeating the sequence, so that the robot is able to establish a new model for it.

3. Feature Extraction for View-Variant Observation

The images recorded in a normal indoor environment contain large amounts of information, but most of it is not relevant for the targeted human motion. Therefore, the first step in vision-based recognition of human actions is to extract relevant information which can be used for further processing. As discussed in Section 2.2.3, this step is to detect and track the subjects of interest in the images, represent their motion in an appropriate form. It is a very important and also difficult step for an intelligent system because that the motion representation is an abstraction of the sensory data, which should catch the features of the motions in the real world, be compact and reliable for later processing. Once the representation has been fixed, the following step to perform a comparison so that classification or recognition can take place. If the representations of the different actions have much ambiguousness, it is hard to achieve successful classification of the observations.

In our work, the attention is focusing on the recognition of human's object manipulations on a table top. These actions could be "take a cup", "put sugar into cup" or a concrete task like "prepare coffee". Since an action takes place in 3-D, and is projected onto a 2-D image, the projected 2-D trajectory may vary depending on the viewpoint of the camera. This variance poses a big problem in the representation and interpretation of hand trajectories. In most current works on action recognition, the issue of view-invariance has been ignored or a fixed view-angle is assumed [34]. However, the assumption does not hold in our scenario because it is costly and unnatural to replicate the point of view of a mobile robot to the human actions throughout all observations. Besides the problem of the multi-view-angle, the choice of features must take the features of hand motion in object manipulations into consideration. The manipulative gesture is different to the face-to-face interactional gesture because the former reflects the interaction between the human's hand and the objects while the latter is typically characterized by a meaningful trajectory of the pure hand movement, e.g. the American Sign Language [114].

This chapter is dedicated to the design, analysis, and extraction of the feature vector for our manipulative action recognition under multi-view observation and the relevant low-level image processing. Firstly, an overview of the feature definitions in the work of multi-view gesture recognition is presented, which leads to the statement of our approach. After that, the computation for locating the hand and objects in the images will be explained. The third section puts forward the construction of the interaction feature vector and the analysis showing its properties.

3.1 Gesture Recognition with Different View-Angles

A single viewpoint of the camera is commonly assumed in gesture recognition because in situations such as giving commands and sign-language recognition, the active user can face the communication partner – the computer – directly. In passive, more observational scenarios like smart rooms and visual surveillance, the human actions and gestures can be observed from different view-angles. Therefore, the approaches to solve the problem of view-angles are dedicated to achieve a *view-invariant* recognition of human actions. Our scenario is similar to the latter but has its own character because of the platform – a mobile robot. It indeed will observe the human from different view-angles. But the capability of a 360° view-invariant recognition is also redundant because, at first, it is not necessary to recognize the human action from the back, secondly, the robot can move to a position where it has a good view of the partner when it wants to know something. From this viewpoint, we will first discuss the approaches for view-invariant action recognition and then present our approach to the problem of view-angles in the following subsections.

3.1.1 Related Work

From a computational perspective, actions are best defined as four-dimensional patterns in space and time. Weinland introduce Motion History Volumes (MHV) as a free-viewpoint representation for human actions in case of multiple calibrated, and background-subtracted, video cameras [125]. The MHV is an extension of Bobick's and Davis's Motion History Images (MHI) from 2-D to 3-D and is represented in cylindrical coordinates. The motion descriptors are the vectors which are formed by concatenating the Fourier magnitudes over Azimuth for all height and radius. The system has achieved good results on the recognition of 11 body motions like "sit down" and "walk". A similar action representation Volume Motion Template (VMT) is put forward by Roh [106]. In his work, the disparity maps from a stereo camera add the depth information of the motion to the 2-D images. The gesture classification is done by the least square distance measurement to the templates.

In principle, many possible sets of features contain enough information to reconstruct the original gesture and be adequate for recognition. To find the optimal feature choice, Campbell evaluated different feature vectors in a T'ai Chi gesture recognition task given translation and rotation of the camera relative to the performer[17]. The raw data (x, y, z) in a 3-D world coordinate are received from a stereo vision system. The feature vectors are derived from the raw data, such as the Cartesian velocity (dx, dy, dz), polar velocity with angular velocity $(dr, d\theta, dz)$, etc. The elements in the feature vectors are tracked over time and the gestures are modeled by HMMs. The work came to the conclusion that

the feature vector $(dr, d\theta, dz)$ had the best overall recognition rate. An interesting result is that the Cartesian velocity (dx, dy, dz) which is not designed to be rotation invariant achieved 77% accuracy on the rotated data despite 45° shift of the observational angle.

The approach of 2-D recognition rather than using 3-D information is motivated by Johannson's experiment with Moving Light Displays (MLD) [56]. In the experiment, people were outfitted in black in a dark background with light attached to their joints. The observer were unable to recognize a human in the image based on the set of static lights. But when the people started moving, the observers were able to discern the presence of a human and recognize its actions. This experiment indicates that the observer can recognize the action relying purely on the motion information and reconstruct a model from 2-D data unconsciously. To recognize actions in video sequences, Rao use dynamic instants as view-invariant features [102]. For each dynamic instant in the trajectory, frame number, location of the hand and "sign" of the instant are stored. The matching is performed on the trajectories with the same number of dynamic instants and same sign permutations. It is highly dependant on the segmentation process. Bashir et al. formulated the view-invariant trajectory representation as an open-end shape representation problem which has a wealth of recent work involving affine-invariant image shape description [5]. He derived two affine-invariant representations for motion trajectories based on Curvature Scale Space (CSS) and Centroid Distance Function (CDF). The first approach using CSS representation detects the CSS contour maxima and codes the trajectories based on the locations of these peaks by HMMs. The limitation of this approach is that it does not model the data between segmentation points. This shortcoming is alleviated in the CDF+PCA based representation scheme. It is a subtrajectory-based representation. For segmentation, the discontinuities in the trajectory are detected with the help of velocity and acceleration. Then, the centroid distance function from the subtrajectory data is computed. The PCA coefficients of the segmented data are then stored to train the HMMs. It outperformed the CSS representation in the experiment on the recognition of the Australian sign language. The data set contains the systhesized trajectories according to the observational view-anlge ranging from $-60°$ to $60°$.

3.1.2 Our Approach

As described before, there are two aspects in selecting features for the perception of the action in our work. The first is that the actions will be observed from different view-angles. However, the qualitative perspective of a robot with regard to manipulative actions performed on a table top (as described in Section 2.2.2) can be assumed to be roughly stable, if the robot is able to choose an appropriate position relative to the human actor with the help of navigation and human detection module. The second is that the actions are not self-explanatory trajectories. The object context must be taken into consideration in the manipulative gesture recognition. Because of these features, a 2-D approach is chosen by us. The reason why we are not using a 3-D representation is two fold. On one hand, the 3-D approaches suffer from the tracking problem in mono-camera images. It is still a field of active research [112]. Better tracking results can be achieved by using stereo cameras, which poses further constraints on the hardware setting. On the other hand, for object manipulations, the representation of the relative motions between the hand and the objects won't have much variance as the projected trajectories which are represented

by the absolute positions of the hand on 2-D images. The limited view-angle also poses a restriction on such variance.

In experiment of Campbell on the evaluation of different feature vectors, the results on Cartesian velocity (dx, dy, dz), that 77% accuracy is achieved on the rotated data with 45° rotation despite that the feature was not designed to be rotation invariant, show a good evidence that the variance of the velocities caused by the different view-angles can to a certain extent be handled by the HMMs. In our approach, we don't use the velocity of the hand but a set of features to describe the relative motion between the hand and objects. The details of the feature vector will be described in Section 3.3.3.

When the trajectories of an action observed in 2-D images from different view-angles are coded by one model, the variance comes not only from the different projections of one trajectory but also the variant performance of the same action over the repetitions. Inspired by the work of Wilson [127], we think the difference of the trajectories of the same action from different viewpoints can also be considered as a kind of systematic error. Wilson used parametric HMMs (PHMM) to model the gestures with the same meaning but different scalar quantity, like the gesture accompanying "this" within the sentence "I caught a fish. It is *this* big". Instead of using a time-invariant linear model for the observation probability, we use a dynamic scaling parameter of the observation model in order to cope with nonlinear changes of the trajectories caused by different view-angles. The details of the approach will be presented in Chapter 4.

3.2 Low-Level Image Processing

In order to represent the relative motion between the hand and the objects, the human hand and objects must be detected. In the following of this section, it will be explained how the human hand is detected and tracked over time. The approach used for detecting the objects in the scene will also be introduced. There is no tracking process on the objects because we focus on the hand motion rather than reconstruct the object motion under severe occlusion during the manipulations.

3.2.1 Hand Detection and Tracking

Today, there exists a variety of different methods for extracting hands from images, each with specific advantages and limitations. The cues commonly used can be grouped into three categories: motion, shape, and color. Each of these cues has certain strengths and weaknesses that make it suitable for specific applications. For example, the motion cue does not only detect moving hands but also the motions of manipulated objects as well as any non-static objects in the background. It is therefore only useful in domains with static background. The shape is only a good cue if the hand that executes the motion exhibits a stable visual structure during the motion. However, human hands usually exhibit a wide range of shapes during interacting with the environment. Only if the hand shape variations are limited, e.g., if most of the fingers are visible during the motion, an adaptive hand-shape tracker could be used [4]. The color cue is suitable for feature extraction in arbitrary domains with respect to hand shape variations and motions in the background, as it is rotation and scale invariant as well as motion independent. However, the major

challenges to a color-based hand extraction is the variations in lighting conditions. Even in a room without windows, different lighting conditions are encountered at different positions in the room. In a typical office environment with the different intensity of the light passing through the window during the day, this becomes even worse.

An adaptive skin-color segmentation algorithm developed by Fritsch is utilized by us for hand detection in a color image sequence [33]. The skin-color is represented in normalized color space, that is obtained by removing the luminance from the color representation through normalization of the individual RGB values:

$$r = \frac{R}{R+G+B} \quad g = \frac{G}{R+G+B} \quad b = \frac{B}{R+G+B} \tag{3.1}$$

As the value for b can be calculated based on the values of r and g with $b = 1 - r - g$, it does not contain additional information and this color space is therefore also referred to as *r-g color space* or *chromatic color space*. According to the work of Störring et al. on the properties of skin color in faces of different ethnical subjects under changing lighting conditions [116], it is shown that the area occupied by the skin color distribution of all different skin types under all possible lighting conditions occupies a shell-shaped area in the normalized color space. This area can be modeled by two quadratic functions [113] and is referred to as the *skin locus*. Using a measured skin locus allows us to realize a preprocessing step in skin color segmentation by discarding all pixel values that are not contained in the measured skin locus.

Consequently, we use Gaussians to model skin color distributions. The skin likelihood for a pixel with color value $\mathbf{x} = (r, g)$ can be calculated for a Gaussian $G(i)$ with mean $\boldsymbol{\mu}_i$ and covariance $\boldsymbol{\Sigma}_i$ using:

$$p_i(\mathbf{x}) = \frac{1}{\sqrt{2\pi \det \boldsymbol{\Sigma}_i}} \exp\left\{-\frac{1}{2}[\mathbf{x} - \boldsymbol{\mu}_i]^T \boldsymbol{\Sigma}_i^{-1} [\mathbf{x} - \boldsymbol{\mu}_i]\right\} \tag{3.2}$$

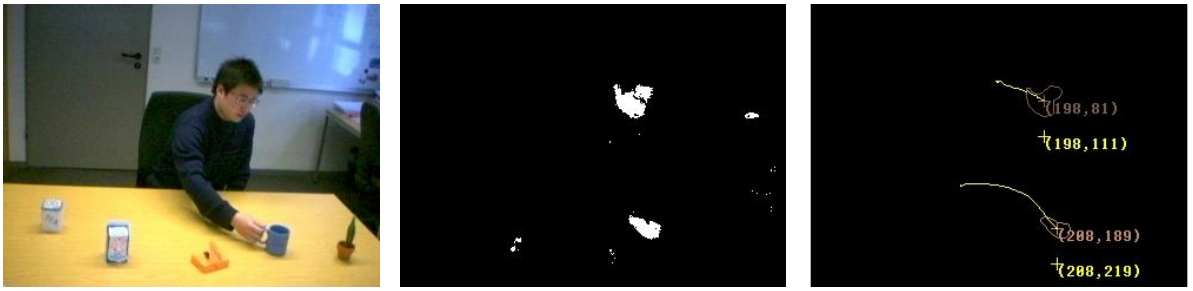
To achieve real-time performance, an unimodal Gaussian is used for each individual skin-colored area. The overall probability of a pixel is calculated as the maximum of the skin likelihood of the individual Gaussians from 3.2:

$$p(\mathbf{x}) = \max_i p_i(\mathbf{x}) \tag{3.3}$$

As a result, it provides for every pixel its skin likelihood for the input image. This probability image is binarized using a classification threshold S_{class} to obtain a label image containing skin and non-skin pixels. The threshold is set to classify a fraction of 98.5% of the training pixels correctly and ignore spurious outliers contained within the last 1.5% of the training pixels, which has been determined empirically.

$$Pr(Y > S_{class}) = 0.985 \cdot N_{train}, \qquad Y = p(\mathbf{x}) \tag{3.4}$$

To remove isolated pixels classified as skin and provide a more homogeneous result, a median of size 5×5 is applied to smooth the label image. Next, a connected components analysis is carried out in the segmentation step to obtain the region segmentation result.



(a) (b) (c)

Figure 3.1: The screen shots from hand detection and tracking

In every frame, the skin-colored regions that exhibit motion will be segmented out. For the objects like hand blob with unknown or flexible shape, the segmented region will be expanded to an area two times the size as before and be used as the region for updating the skin color model. The hand blob is tracked over time using Kalman filtering [107]. The model of the Kalman filter represents the kinematic motion equation of one point using a constant acceleration model. The center of mass (COM) of each region is tracked individually. Figure 3.1 shows the screen shots of the processing. Figure 3.1(a) shows the raw image. Figure 3.1(b) is the thresholded image indicating the skin-color pixels. Figure 3.1(c) displays the trajectories from skin colored area tracking. Currently only single hand manipulations are assumed. Therefore, the bigger skin-color region is labeled as face. The smaller is the hand. The hand observation $\mathbf{o}_t^{\text{hand}}$ is represented by the hand position (h_x, h_y) at time t.

$$\mathbf{o}_t^{\text{hand}} = (h_{x,t}, h_{y,t}) \tag{3.5}$$

3.2.2 Object Information Acquisition

To represent manipulative gestures by relative motions between hands and objects, not only the human hand but also the individual objects must be extracted from the images. Therefore, a reliable detection of objects is crucial for the overall system performance.

The field of object recognition has seen tremendous progress over the past decades, both for specific domains such as face detection [121], and for more general object domains [120]. But it is also fair to say that the general solutions to object recognition is still far from the current state. The challenges are coming from many aspects. For the individual object instance, its appearance in 2-D images is affected by the distance and the view-angle of the camera. Creating invariance to scaling, translation, rotation is a big problem. The situation gets even worse when it is partially occluded. Note that some subjects are not invariant at all, especially w.r.t. 3-D rotations. Moreover, objects are not generally presented against a neutral background, but are embedded in clutter. When the recognition goes to the category layer, representing the huge variance of the objects in the same class like "lamp" is still an open question to the computer vision researchers. An overview of the object recognition is not the focus of our work. For this purpose, the text books [32; 23; 108] and recent papers are good references.

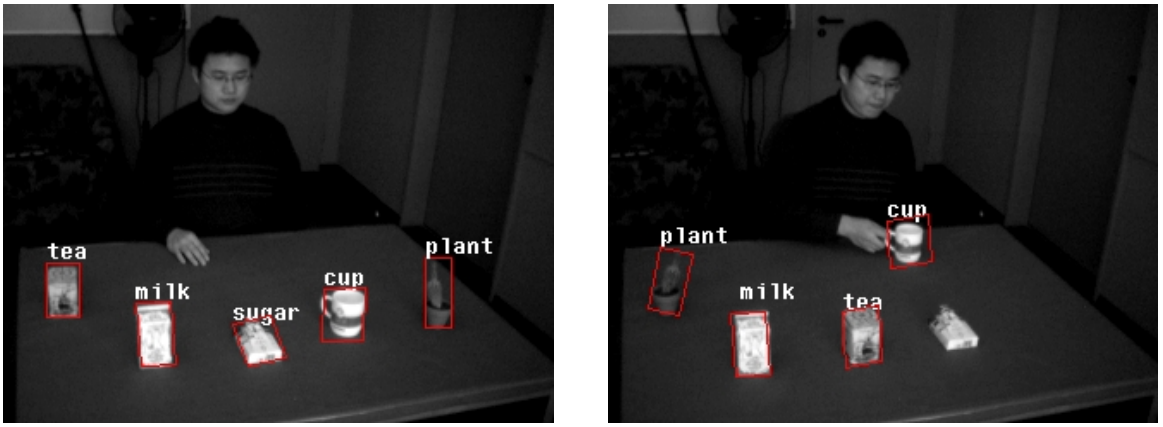


Figure 3.2: Object detection using SIFT feature

In order to avoid partial occlusion problems with interacting hands and achieve the size of the objects in the scene, we use a Scale Invariant Feature Transform (SIFT) [75] based object recognizer in our system. The scale-invariant features are efficiently identified by using a staged filtering approach. The first stage identifies keypoint locations in scale space by looking for locations that are maxima or minima of a difference-of-Gaussian function. Each point is used to generate a feature vector that describes the local image region sampled relative to its scale-space coordinate frame. The SIFT keypoints derived from an image are used in a nearest-neighbor approach to index candidate object models. Collections of keypoints that agree on a potential model pose are first identified through a Hough transform hash table, and then through a least-squares fit to a final estimate of model parameters. When at least 3 keypoints agree on the model parameters with low residual, there is strong evidence for the presence of the object. By grouping the scales of the matched keypoints, the sizes of the objects in the scene also can be estimated. But this method generates very few or no keypoints if the objects are very plain and do not have much detail. So we simplified the scenario by using the objects with obvious textures.

It is supposed that the detector is applied on the static scene because it will be costly to track the object during the manipulations which cause severe occlusion of the object in hand. Consequently, the relative motion is purely defined based on the hand trajectory that approaches an object instead of considering the object-in-hand context in which the object is being moved. If a moved object is applied to another object, the second object defines the object context.

The observation vector of a detected object $\mathbf{o}_i^{\text{obj}}$ contains its position (o_x, o_y), a unique identifier (ID) for each different object type in the scene and its height o_h and width o_w. As we can have several objects in the scene, the overall object observation vector contains multiple objects:

$$\mathbf{o}^{\text{obj}} = \{\mathbf{o}_1^{\text{obj}}, \ldots, \mathbf{o}_i^{\text{obj}}, \ldots, \mathbf{o}_L^{\text{obj}}\} \tag{3.6}$$

with

$$\mathbf{o}_i^{\text{obj}} = (o_x, o_y, \text{ID}, o_h, o_w) \tag{3.7}$$

Figure 3.2 shows the results of object detection using SIFT feature. There are two gray images because the SIFT feature is calculated based on gray images. It can be found that in this scenario the objects were robustly detected with correct size and orientation. However, no object detector is perfect. In the left image an deletion error is also presented.

3.3 Feature Vector Construction for Manipulations

Even if the hands and objects are detected from the images, their positions in images coordinate can not be used directly for recognition. This section will firstly present the object-oriented strategy chosen for the motion representation. Then, a concept named object vicinity is introduced by us to avoid the unnecessary processing on the meaningless trajectory and scale the feature vectors. After that, the definition of computation of the individual elements of the feature vector will be explained. A comparison between our feature vector and traditional ones will be given at the end of this section.

3.3.1 Object-Oriented Approach

There are three main approaches to couple the hand trajectories and symbolic object context in the image: hand-centered, object-centered, and parallel processing. The hand centered approach is put forward by Fritsch [34]. The basic idea is that a trajectory template gains more weight if the expected object is found in the searching area which is predefined for the template. In the work of constructing task models for manipulations, Ogawara represent the motion in the target object's coordinate frame, i.e., in an object-centered way [87]. The objectspace introduced by Moore is a hierarchical framework which integrates preknowledge and different kinds perceptions, such as object and motions, together in a parallel way [80]. The multicues are fused by a Bayesian network to infer the human activities. Another example of parallel processing is the work from Yu [135]. Parallel hidden Markov models (PaHMMs) are applied to integrate the object in the direction of eye gaze and hand movements for task recognition.

The proposed approach by us is called *object-oriented* w.r.t. two different aspects: it is *object-centered* in terms of trajectory features that are defined relative to an object, and it uses *object-specific* models for action primitives. Each object has its own set of manipulative primitives because we argue that different object types serve different manipulative functions and even manipulations with the same functional meaning are performed differently on different objects. Here we take the idea from Moore that the objects can be organized familiar to classes in the object-oriented programming language. Object types can be thought as *classes*. Then, a concrete object instance could correspond to an "object" of one class with its own properties. The relationships between the object types and different layers for description can be managed by the communication and inheritance between the classes.

3.3.2 Object Vicinity

For manipulative gesture recognition, the concept of *object vicinity* is introduced by us. Its functionality is two-folded. On one side, it can decrease the system load by getting rid of the processing on the trajectories where the hand is still far from the objects because

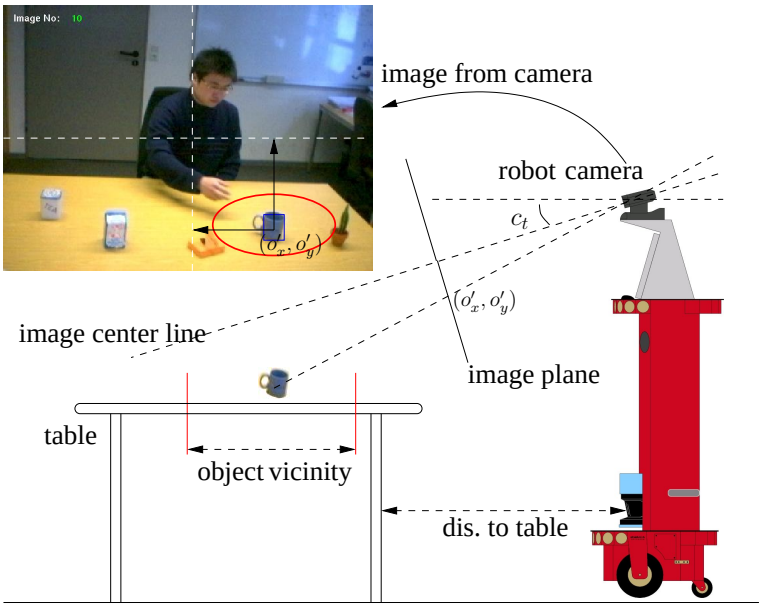


Figure 3.3: The projection of an object vicinity on a 2-D image

it is common sense that the relative movement between hand and object contains less interaction features when they are far away from each other. On the other side, the motions in the vicinities of different objects can be unified when they are scaled with regard to the size of the vicinities.

The vicinity of an object can be defined in images or in the real world. But the correct distance of two subjects in the real world can not be deduced from their distance in 2-D images without depth information. The object vicinity is defined in real world. In our scenario, the manipulations will mainly happen on a flat table surface. Therefore, a vicinity of an object on the table is defined in the 2-D plane rather than in 3-D space. It is centered in the middle of the object and limited by the ratio β of its radius and the object size, which is shown in Figure 3.3. One point in this definition that we need to specify is how we can get the projection of it in 2-D images without a complete 3-D representation.

Because the vicinity of an object is relative to the object size in the real world, the projection of the objects come to consideration first. To project a point from real world to an image plane, the position of the point in the camera coordinates and the intrinsic calibration parameters of the camera must be known. The contour of the object projections will vary according to three aspects: the view-angle of the camera, the object-camera distance and object poses on the table. In our system, there is no precise 3-D models of the objects. The object poses and their distances to the camera are also infeasible. Therefore, we assume that the size of an detected object in the image is inverse proportional to its distance to the camera.

$$r_{\text{obj}} = \mathbf{C} \cdot \frac{1}{d_{oc}} \tag{3.8}$$

C is a constant value, r_{obj} is the object radius in the image and d_{oc} is the distance between camera and object. This assumption is equal to the situation that the camera is always observing a sphere which is on the optical axis of the lens.

To achieve the projection of the border of the object vicinity, further conditions are needed. Despite the assumption on the inverse proportion between the object size in image and its distance to the camera, the table surface can also pose differently in the camera coordinate. On the robot platform, the tilt angle of the robot camera c_p is known in realtime. Plus that a flat table surface is assumed in our scenario, the round boundary of a vicinity will be projected into a closed curve in 2-D images, Figure 3.3 illustrates geometry of the projection. Note that the closed curve is not an ellipse with the object in the center. In order to avoid a complex representation of the curve caused by the 3-D to 2-D projection, it is approximated to by an ellipse. It is centered by the detected object. The lengths of the axes of the ellipse for object i is calculated as

$$\begin{aligned} a_i &= r_i \cdot \beta \cdot \cos(\arctan(\frac{o'_x}{f})) \\ b_i &= r_i \cdot \beta \cdot \sin(c_p + \arctan(\frac{o'_y}{f})) \end{aligned} \tag{3.9}$$

In it, a_i and b_i are the horizontal and vertical semi-axes. r_i is the radius of the object i in the image. o'_x and o'_y are the offset of the object position to the image center. f is the camera focus measured in pixels. This approximation is derived from the above assumption on object size. The first assumption is equal to the situation that the object is on the optical axis of the lens. Therefore, the optical axis of the camera is "moved" to the center of the detected object. The direction of the object can be computed by combining the camera tilt angle and the object position in the image. The vicinity is then projected as if the optical axis would go through the detected object. By applying the approximation, the pre-knowledge for achieving the vicinity of an object in 2-D images only consists of the tilt angle and intrinsic calibration parameters of the robot camera. There is no need to know the distance between the robot and the table, the height of the table, the height of the robot, etc., which gives great flexibility to the system.

Based on this vicinity, a pre-segmentation step of the hand trajectory is performed that ignores irrelevant motions for primitive recognition. Considering the possible occlusions in manipulation and the uncertainty in moving an object, a segment is started when the hand enters the vicinity or when an object is detected and the hand is already in the vicinity (object put down into the scene). It ends when the hand goes out of the object's vicinity or when the object is lost after the hand moves away (object has been taken). As a consequence, the trajectory is segmented differently based on the different objects in the scene. To handle this multi-observation problem, one processing thread is started for each detected object. In the next chapter, the processing of a single thread will be introduced. There, the final segmentation is directly coupled with the recognition step.

3.3.3 Feature Definition and Evaluation

Because of the object-centered trajectories representation, the features of the hand motions are different with regard to the different objects in the scene. In the light of work of

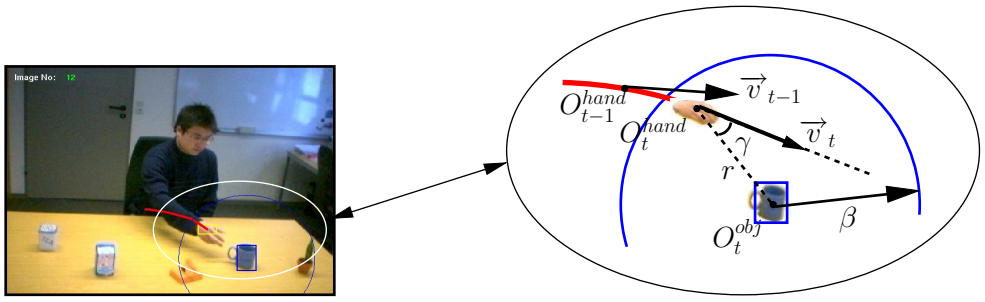


Figure 3.4: The illustration of feature vector

Campbell [17], we choose the motion descriptors which reflect the relative motion between hand and object to construct the feature vector. Furthermore, considering view-variant problem, an approximate 3-D to 2-D projection of the object vicinity was put forward in the previous section. The shape of the ellipse indicates the distortion of the 3-D trajectory because of the projection. Therefore, if the distortion is considered for extracting the motion features from 2-D trajectories, the real features of the motions in 3-D will be to a large extent preserved.

In each object detected in the image, the interaction of the hand and the object is represented by a five-dimensional feature vector $\mathbf{v}_i^f$ that is calculated from $\mathbf{o}^{\text{hand}}$ and $\mathbf{o}_i^{\text{obj}}$. It contains: distance d_i between the object and the operative hand, magnitude of hand speed v_i, change of hand speed, Δv_i, change of speed direction $\Delta\alpha_i$, as well as the angle γ_i of the line connecting object and hand relative to the direction of the hand motion.

$$\mathbf{v}_i^f = (d_i, v_i, \Delta v_i, \Delta\alpha_i, \gamma_i) \tag{3.10}$$

The hand-object distance d_i is not the absolute distance in the images but measured with regard to polar coordinates scaled by object size. Suppose that hand position relative to the center of the object vicinity is $(h'_{x,i}, h'_{y,i})$,

$$d_i = \sqrt{\frac{{h'_{x,i}}^2}{{a_i}^2} + \frac{{h'_{y,i}}^2}{{b_i}^2}}, \tag{3.11}$$

The magnitude of hand speed v_i is the substraction of d_i of two successive time step. The change of the hand speed Δv_i is then the first order derivative of v_i. The change of speed direction $\Delta\alpha_i$ is calculated based on the last three hand positions. It has the sign "positive" when the hand turns to the side where the object is and "negative" when other way around. The hand-object angle γ_i is the included angle between current hand motion direction and the line connecting the object and the hand.

This feature vector is NOT view-invariant. But as presented in Section 3.1.2, instead of seeking for view-invariant features, our approach pursues a representation of the gesture which has small variance when a gesture motion is observed from different view-angles. When we have a close look at the individual components, these components can be sorted into two categories–distance and angle. The first category includes $d_i, v_i, \Delta v_i$ which are

calculated based on the distances between the consecutive points of the hand. By introducing the object vicinity, the values of these features of the same action perceived from different view-angles can be transformed into a normalized and comparable feature space. In practice, Δv_i is discretized into 2 discrete states ≥ 0 and < 0 indicating acceleration and deceleration of the hand and keep d_i and v_i continuous to achieve a balance between the precision and the demanding on the large mount of training data. The second category consists of $\Delta\alpha_i, \gamma_i$. They are not affected by the object vicinities and have a large variance with respect to view-angles. Therefore, the change of speed direction $\Delta\alpha_i$ is discretized into 2 discrete states ≥ 0 and < 0 indicating whether the hand turns to the side of the object or not. The hand-object angle γ_i is discretized into 2 discrete states $[0^\circ \ldots 90^\circ]$ and $(90^\circ \ldots 180^\circ]$ which means the hand moves towards the object and away from the object individually. Concluding the manipulations on all components, the effect of the view-variant observation on the feature vector is depressed.

Figure 3.5 shows the effect using simulated data. Two trajectories are from the same action in reality, which is 50cm long and performed horizontally on the table surface with constant acceleration, but observed from two view-angles with 90 degree difference in horizontal angle and the same pitch angle 20.5°. Figure 3.5(a) display the trajectories and objects in a 3-D space. Figure 3.5(b) shows their projected trajectories in an 2-D image coordinate with size 320x240. Figure 3.5(c) is the original distance vs. speed display measured in pixel and Figure 3.5(d) presents the same information in the transformed feature space. Figure 3.5(e) shows continuous relative angles at different time step. In Figure 3.5(f), the relative angles are discretized into 2 discrete states.

To show whether the defined feature vector is useful to decrease the variance caused by the view-angles, the similarity of the trajectories before and after the space transformation must be measured. The L_p norm is a popular point-to-point measure of distance, and it is defined for two m-dimensional points a and b as

$$L_p(a,b) = \left[\sum_{i=1}^{m} |a_i - b_i|^p\right]^{1/p} \tag{3.12}$$

Setting p to 1 gives the Manhattan distance, and the Euclidean distance is obtained when $p = 2$. In order to measure the similarity between two trajectories in different scales, we extend the L_p norm to measure the similarity of two trajectories $traj_1$ and $traj_2$:

$$L_p(traj_1, traj_2) = \frac{1}{N}\sum_{j=1}^{N}\left[\frac{1}{m}\sum_{i=1}^{m}\left(\frac{|traj_{1,ji} - traj_{2,ji}|}{|traj_{1,ji} + traj_{2,ji}|}\right)^p\right]^{1/p} \tag{3.13}$$

N is the length of the trajectory. When $p = 2$, the L_p of these two trajectories in Figure 3.5(c) and Figure 3.5(d) are 0.5862 and 0.3432 individually. Therefore, the difference of the two trajectories are depressed after the transform.

Except using simulated data, we also test the feature vector on real data. In the experiment, 4 cameras are used. They recorded the performed action simultaneously. Their positions and view-angles are listed in Table 3.1. They have the same height as the robot BIRON [44]. Figure 3.6 shows the pictures captured by the cameras. Figure 3.7 shows the

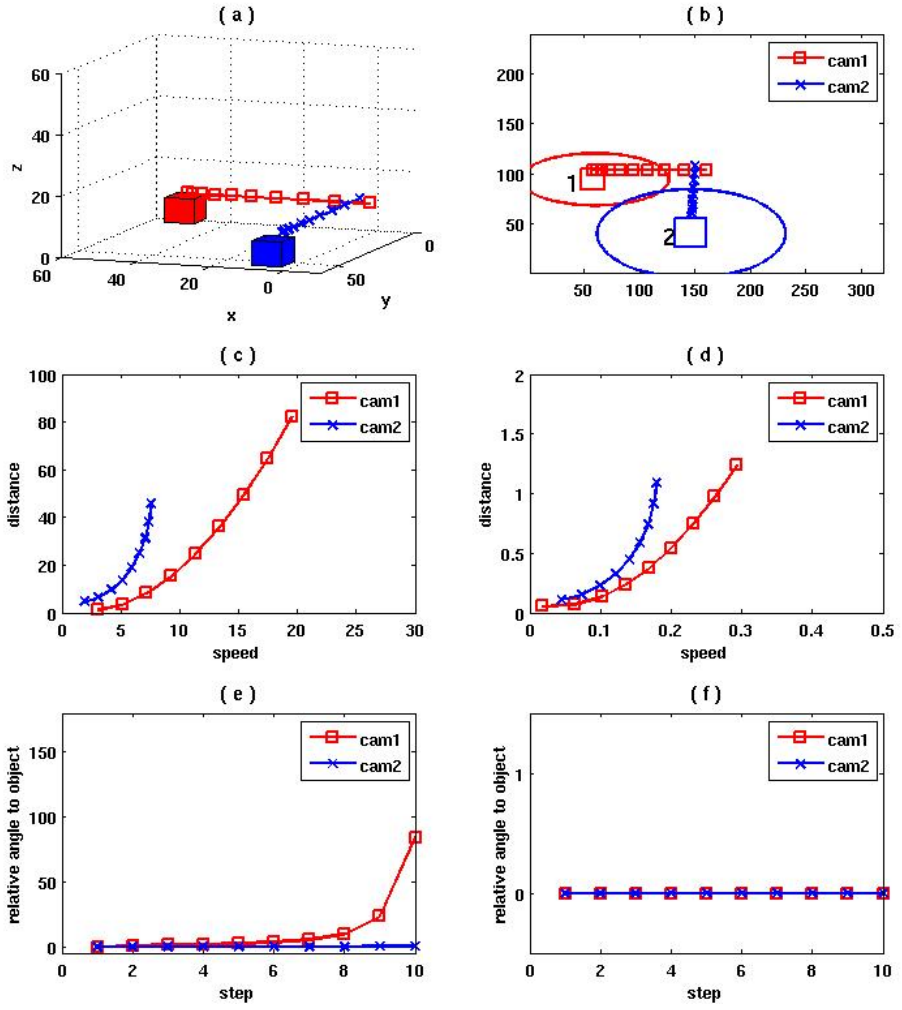


Figure 3.5: The effect of feature transform (a)3-D display of the trajectories (b) projected trajectories in pixel coordinates (c)distance vs. speed measured in pixel (d) distance vs. speed in transformed feature space (e)continuous relative angle vs. time (f)discrete relative angle vs. time

different trajectories of a "take cup" primitive observed from these 4 cameras. On the left column the features of the primitive are measured in pixel. The right column shows the values we used. The L_p of the speed-distance trajectories before and after transformation are 0.4829 and 0.3330. This means for the real trajectories the variance of them is also decreased by using the transformed feature vector.

3.4 Summary

Different to much related work in gesture recognition which assume a fixed static camera view, our system aims to solve a view-independent recognition problem because a mobile robot companion could observe the actions from different view-angles in our scenario. This chapter discussed different approaches which focused on the view-independent gesture

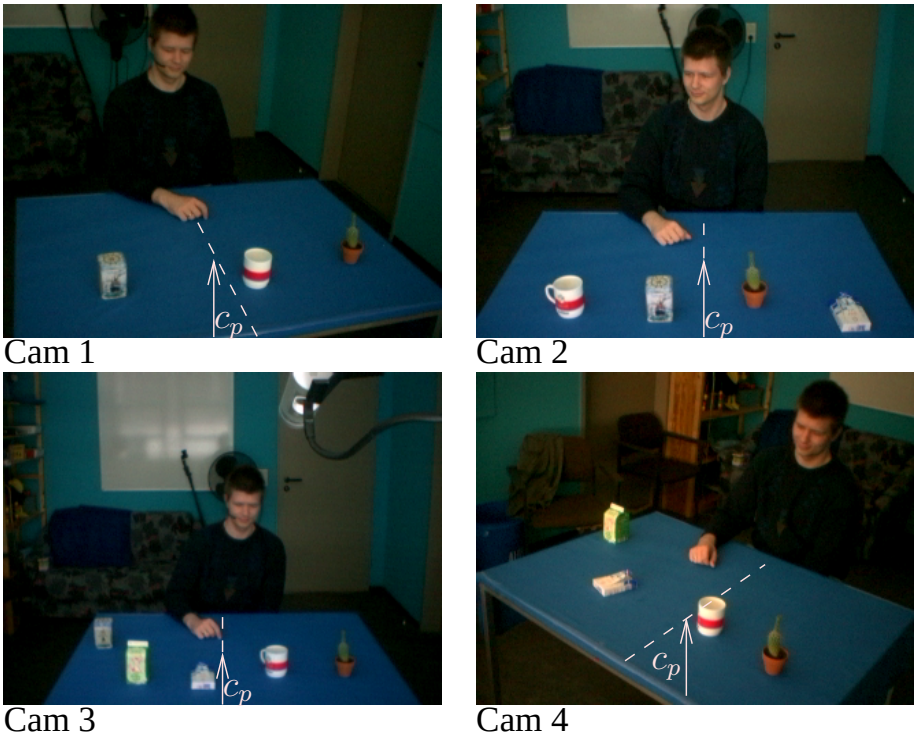


Figure 3.6: Views from 4 cameras

recognition. These approaches can be generally sorted into two categories: 3-D and 2-D approach. After an analysis of the real observation conditions of our robot and also taking the hardware constrains of our robot and into consideration, a 2-D based approach is chosen by us.

More concretely, this chapter presented three aspects towards this direction. Firstly, the low level image processing methods on 2-D images are put forward. The human hands are detected by color-based segmentation and tracked through Kalman filter. The static objects in the scene are detected using SIFT feature based object detector. Secondly, we proposed an object-oriented approach to combine the hand trajectories and object context together. The object-oriented approach has two aspects: it is object-centered in terms of trajectory features that are defined relative to an object and it uses object-specific models for manipulative primitives. This approach gives a clear answer to the question which appears in many other works dealing with object context: when and how

Table 3.1: The positions and view-angles of the cameras

	cam 1	cam 2	cam 3	cam 4
dis. to table (cm)	100	100	150	100
pitch angle ($^\circ$)	17.3	0	0	30.4
horizontal angle ($^\circ$)	21.6	20.5	12.4	22.7

to include object context in gesture recognition. In addition, it also provides an easy way to pre-segment the hand trajectories and decrease the processing load. The last but not least point in this chapter is the design of the feature vectors for the manipulative gesture recognition. We used an interactive feature vector that represents the relative movements between the human hand and the objects. According to the different appearance of the features in view-variant observation, the feature vector are designed as semi-continuous. Instead of measuring the features directly in pixel of the images, the features are scaled by the object vicinity. The concept of object vicinity comes from the idea that more far away is the hand from the object, less connectivity exists between them. The space of object vicinity in 2-D images is calculated only based on the camera tilt and object detection results. The distance between the robot and the table, the height of the table, the height of the robot, etc. are not needed, which gives great flexibility to the system. The advantage of the scaled feature vector over original measurements in pixel is proofed by using both simulated data and real data.

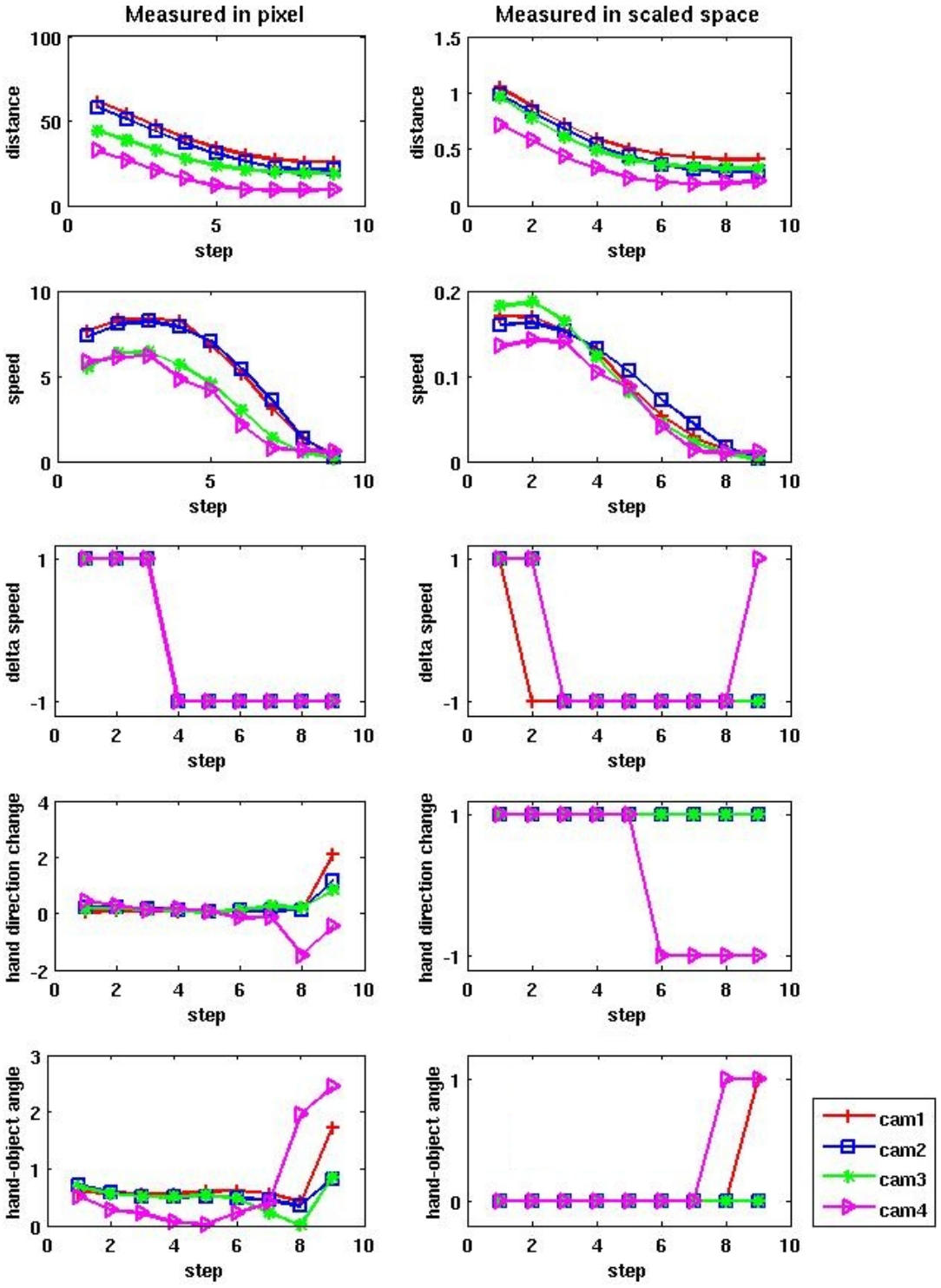


Figure 3.7: The effect of feature transform shown by real data

4. Manipulative Primitive Detection

The previous chapter presented how the feature vectors describing the manipulative gestures are extracted from the images. observed motion. The next step for recognizing the action is to interpret the trajectory. Here the symbol grounding problem arises (see Section 2.2.3). We choose a two-layer approach – manipulative primitive and manipulative task, which is motivated by neuroscientific evidence on motor primitives – a set of movement programs that form a vocabulary for the generation of a variety of complex movements [94; 104]. A manipulative task is made of different elementary manipulations on individual objects. For example, "water a plant" consists of "take a cup", "pour water into the plant pot", and "put down the cup".

As an object-oriented approach (see Section 3.3.1), the manipulative primitives are typical hand operations on objects, for example, a cup can be taken – "take cup". But how can we correctly detect the primitives from a continuous performance? Although an object vicinity is defined for cutting away the hand trajectories which are less relevant to object manipulation (see Section 3.3.2), it is a rough segmentation and mainly used for decreasing the processing load and providing a normalized space for the features (see Section 3.3.3). The relative movements of hands in an object vicinity or part of them could also be meaningless. Therefore, the trajectory segmentation/spotting problem will be discussed together with the matching methods in this chapter.

As mentioned in Section 2.2.3, there are two main aspects affecting the modeling and recognition: the human performance and the image perception process. The former includes aspects like variations in the repeated performance of the same activity even for the same person. Different individuals perform similar activities in significantly different ways. In this context, defining the onset and offset of an activity is challenging as similar activities frequently have different temporal durations. The latter contains issues like occlusion during performance as well as variant viewpoints and observational distances. To cope with the trajectory variance coming from the human performance, a lot of methods have been put forward, such as dynamic time warping [1], artificial neural networks [60], and hidden Markov models [101]. For the trajectory variance caused by the variant camera view-angle, Section 3.1.1 gave an overview of the solutions. But to our knowledge,

there are no framework which takes both aspects – human performance differences and the perception differences into account. We novelly differentiate these variance into random and systematic "errors" and integrate them in an unified graphical model [71].

In this chapter, we will present our framework to detect the primitives from the view-variant observations, which is the implementation of the idea in Section 3.1.2. Firstly we briefly describe the applications which are dealing with the elementary trajectory recognition and trajectory spotting in the related work. Then, two typical trajectory matching methods and their extensions for trajectory spotting will be introduced. Based on an analysis of these techniques, the novel algorithm named particle filter realized hidden Markov model matching (PF-HMM) is put forward. In this section, it is explained in detail how it can be used for the detection of the primitives which contains the different types of variances. At last, we evaluate the proposed algorithm in an experiment and give a conclusion of it.

4.1 Related Work

Because of a wide range of applications of the human motion recognition in visual surveillance, advanced user interfaces, etc., the trajectory matching methods have been investigated for a long time. There are two main approaches: template matching and state-space methods. In the template-based matching methods, the templates used are quite different to each other. Bobick and Davis made use of the binary motion-energy images (MEI) and motion history images (MHI) as two components in a temporal template. The recognition is done by measuring the distance of the moment statistical description between input MEI and MHI and the templates [10]. In order to achieve the online segmentation, an exhausting minimum distance search goes backward through the interval between the minimum and maximum duration of the action. Waldherr used temporal templates to model the command gestures to the robot [124]. Gesture templates are composed of a sequence of feature vectors, constructed from a small number (e.g., 5) of training examples. The temporal template matcher matches the gesture template to the most recent n feature vectors. For varying numbers of n, in their implementation, $n = 40, 50, \ldots, 80$, the Viterbi algorithm was used for time alignment. The Viterbi alignment employs dynamic programming to find the best temporal alignment between the feature vector sequence and the gesture template. The advantage of using template matching technique is its inexpensive computational cost and simplicity in implementation. However, it is relatively sensitive to noise and the variance of the movement duration.

The approach based on the state-space models defines a set of key states and uses certain probabilities to generate connections between the states. Nowadays it has been widely applied to prediction, estimation and detection of temporal series. The HMM is the most representive method used to study sequential signals. It is effectively used in speech recognition [50], handwriting recognition [126] and human activities recognition [25; 74; 90]. An early work by Starner and Pentland concentrated on recognizing the gestures of hands performing american sign language (ASL) [115]. Their method for recognizing ASL with HMM reaches a recognition accuracy of over 91% on a set of 40 gestures. This rate can be increased to above 99% by using a grammar for ASL sentences. In their work, the

subject wears distinctly colored gloves on both hands and sit in a chair in front of the camera to aid hand tracking.

By defining the transitions between states and the state dependend observations in a probabilistic way, variations can be coped with to a certain degree. However, the standard forward algorithm to calculate the probabilities of the HMM candidates given the observation has the assumption that the whole sequence is emitted by one HMM. In order to spot the partition which conforms to a HMM from a long observation, some approaches, e.g. HMM-based threshold model [65] and normalized Viterbi algorithm [81] were put forward. Because the output score of the continuous observations of a given HMM will decrease monotonouslly, a sliding window is used to tune the weights of the observation. Although the state-space approach may overcome the disadvantages of the template matching approach, its learning usually involves complex iterative computation. Meanwhile, how to select the proper number of states and the topology of the state network remains a difficult issue.

Artificial neural networks (ANN), especially recurrent neural networks (RNN) have also been used for gesture recognition [60]. ANN has achieved good recognition results on static patterns like postures. However, it is not suited for the manipulative primitives having large temporal variance. It also needs a large mount of data for training.

Recently, the sequential Monte Carlo (SMC) method also named particle filter (PF) is getting more and more focus in the pattern recognition society, which allows an on-line approximation of probability distributions using samples (also named particles). It has been used for template-based trajectory matching [6].

In order to keep the spatio-temporal variability of HMMs and use the advantage of PF on tracking the models with weighted particles, a PF realized HMM matching method is proposed to detect object-specific manipulative primitives. This process is building the bridge between the low-level image processing and the task knowledge in the system.

4.2 Elementary Trajectory Recognition and Spotting

Because of the continuous observation in our scenario, the segmentation of the motion trajectories is a necessary step. The gestures are supposed to happen naturally in an indoor environment. There is no clear onset and offset of an elementary trajectory. Therefore the segmentation is coupled with the recognition process. It is also named spotting, which first appeared in the speech recognition society to find keywords in a text. The spotting of the trajectory is more difficult because of the arbitrary form of the signal. There are two main approaches focusing on the trajectory spotting: dynamic programming (DP) based methods and HMM based methods. The strategy for the presentation of these two approaches is that for every approach, firstly the basic matching method for segmented trajectories will be introduced, and then the derivative – the spotting method.

4.2.1 Dynamic Time Warping

One of the earliest approaches to isolated word speech recognition was to store a prototypical version of each word (called a template) in the vocabulary and compare incoming

speech with each word, taking the closest match. Dynamic time warping (DTW) was introduced by Sakoe [109] to match the speech waveforms, whose duration and the strength of each spoken sound are variant. It is a technique that finds the optimal alignment between two time series if one time series may be "warped" non-linearly by stretching or shrinking it along its time axis. This warping between two time series can then be used to find corresponding regions between the two time series or to determine the similarity between the two time series. In addition to speech recognition, dynamic time warping has also been found useful in many other disciplines, including data mining, gesture recognition[1], and gene sequence analysis etc.

The dynamic time warping problem is stated as follows [110] : Given two time series $\mathbf{x}$, and $\mathbf{y}$, of lengths n and m,

$$\begin{aligned} \mathbf{x} &= x_1, x_2, \ldots x_i, \ldots x_n \\ \mathbf{y} &= x_1, x_2, \ldots x_j, \ldots x_m \end{aligned} \tag{4.1}$$

construct a warp path $\mathbf{w}$

$$\mathbf{w} = w_1, w_2, \ldots w_k, \ldots w_l, \qquad \max(n, m) \leq l < n + m \tag{4.2}$$

where l is the length of the warp path and the k^{th} element of the warp path is

$$w_k = (i, j) \tag{4.3}$$

where i is an index from time series $\mathbf{x}$, and j is an index from time series $\mathbf{y}$. The warp path must start at the beginning of each time series at $w_1 = (1, 1)$ and finish at the end of both time series at $w_l = (n, m)$. This ensures that every index of both time series is used in the warp path. There is also a constraint on the warp path that forces i and j to be monotonically increasing in the warp path. Every index of each time series must be used. Stated more formally:

$$w_k = (i, j), w_{k+1} = (i', j') \qquad i \leq i' \leq i + 1, j \leq j' \leq j + 1 \tag{4.4}$$

The optimal warp path is the minimum-distance warp path, where the distance of a warp path $\mathbf{w}$ is

$$d(\mathbf{w}) = \sum_{k=1}^{k=l} d(w_k) \qquad with\ d(w_k) = d(x_i, y_j) \tag{4.5}$$

$d(\mathbf{w})$ is the distance (typically Euclidean distance) of warp path $\mathbf{w}$, and $d(w_k)$ is the distance between the two data point indices (one from $\mathbf{x}$ and one from $\mathbf{y}$) in the k^{th} element of the warp path.

A dynamic programming approach is used to find this minimum distance warp path. Instead of attempting to solve the entire problem all at once, solutions to sub-problems (portions of the time series) are found, and used to repeatedly find solutions to a slightly larger problem until the solution is found for the entire time series. A two-dimensional n by m cost matrix D, is constructed where the value at $D(i, j)$ is the minimumdistance warp path that can be constructed from the two time series $\mathbf{x}' = x_1, \ldots, x_i$ and $\mathbf{y}' = y_1, \ldots, y_j$. The value at $D(n, m)$ will contain the minimum distance warp path between time series

$\mathbf{x}$ and $\mathbf{y}$. Both axes of D represent time. The x-axis is the time of time series $\mathbf{x}$, and the y-axis is the time of time series $\mathbf{y}$.

To find the minimum-distance warp path, every cell of the cost matrix must be filled. The rationale behind using a dynamic programming approach to this problem is the following. The value at $D(i, j)$ is the minimum warp distance of two time series of lengths i and j. If the minimum warp distances are already known for all slightly smaller portions of that time series that are a single data point away from lengths i and j, the value at $D(i, j)$ is the minimum distance of all possible warp paths for time series that are one data point smaller than i and j, plus the distance between the two points x_i and y_j. Since the warp past must either be incremented by one or stay the same along the i and j axes, the distances of the optimal warp paths one data point smaller than lengths i and j are contained in the matrix at $D(i-1, j)$, $D(i, j-1)$, and $D(i-1, j-1)$. So the value of a cell in the cost matrix is:

$$D(i, j) = d(i, j) + \min(D(i-1, j-1), D(i-1, j), D(i, j-1)) \tag{4.6}$$

The warp path to $D(i, j)$ must pass through one of those three grid cells, and since the minimum possible warp path distance is already known for them, all that is needed is to simply add the distance of the current two points to the smallest one. After the entire matrix is filled, a warp path must be found from $D(1, 1)$ to $D(n, m)$. The warp path is actually calculated in reverse order starting at $D(n, m)$. Whichever of these three adjacent cells has the smallest value is added to the beginning of the warp path found so far, and the search continues from that cell. The search stops when $D(1, 1)$ is reached.

The DTW algorithm is able to find the optimal alignment between two time series. However, it is not supposed to find an optimal matching of a template in a longer trajectory. To this end, the method known as Continuous Dynamic Warping (CDP) is put forward by Oka [89].

Continuous Dynamic Warping (CDP): Let $\mathbf{o}$ and $\mathbf{z}$ denote variables to represent input and reference time sequences, respectively. An input time sequence $\mathbf{o}$ is a function of the discrete time t which is infinite:

$$\mathbf{o} = \{f(t) | t = 1, 2, 3, \ldots\}. \tag{4.7}$$

A reference time sequence is defined by $\mathbf{z}$ which represents a template:

$$\mathbf{z} = \{z(\tau) | 1 \leq \tau \leq T\}. \tag{4.8}$$

The parameter τ is bounded as $1 \leq \tau \leq T$, where T is the length of the reference pattern. Another two variables are also needed here, namely, $d(t, \tau)$; a local distance between $f(t)$ and $z(\tau)$, and $D(t, \tau)$; a minimum accumulated value of local distances. The variable $D(t, \tau)$ has initial conditions for the cases of $t = -1$ and $t = 0$:

$$D(-1, \tau) = D(0, \tau) = \infty. \tag{4.9}$$

An iteration rule to update $D(t, \tau)$ defined by:
for $\tau = 1$,

$$D(t, 1) = 3 \cdot d(t, 1), \tag{4.10}$$

for $\tau = 2$,

$$D(t,2) = min \begin{cases} D(t-2,1) + 2 \cdot d(t-1,2) + d(t,2) \\ D(t-1,1) + 3 \cdot d(t,2) \\ D(t,1) + 3 \cdot d(t,2) \end{cases} \tag{4.11}$$

for $3 \leq \tau \leq T$,

$$D(t,\tau) = min \begin{cases} D(t-2,\tau-1) + 2 \cdot d(t-1,\tau) + d(t,\tau) \\ D(t-1,\tau-1) + 3 \cdot d(t,\tau) \\ D(t,\tau-2) + 3 \cdot d(t,\tau-1) + 3 \cdot d(t,\tau) \end{cases} \tag{4.12}$$

The iteration is based on DP to optimally accumulate local distances along a locus on the (t,τ)-plane from the line with $t = 1$ to the line with $\tau = T$ giving the minimum accumulated value. The CDP output, $A(t)$, is defined as follows:

$$A(t) = \frac{1}{3 \cdot T} D(t,T). \tag{4.13}$$

The constant value $3T$ becomes a coefficient parameter to normalize the value of $D(t,T)$ to be compared with corresponding values of other reference patterns with different length. A segmented part of an input sequence is obtained each time if the value $A(t)$ gives a local minimum below a threshold value.

In brief, continuous DP finds the optimal path and minimum accumulated distance by choosing the minimum local path successively from bottom left to top right in the reference-input plane. The $D(t,T)$ is the distance between the whole reference and the input considering temporal warps from 1/2 to 2 times. It has a quadratic time and space complexity that limits its use to only small time series data sets.

4.2.2 Hidden Markov Model

Hidden Markov model is a powerful statistical tool for modeling generative sequences that can be characterized by an underlying process generating an observable sequence. HMM have found application in many areas interested in signal processing, and in particular speech processing, but have also been applied with success to the recognition of human motion sequences in computer vision [74; 28; 135].

HMM is represented as a collection of finite states connected by transitions. Each state is characterized by two sets of probabilities: a transition probability, and either a discrete output probability distribution or continuous output probability density function which, given the state, defines the condition probability of emitting each output symbol from a finite alphabet or a continuous random vector. Even if the output symbols are observable, we cannot say exactly what state sequence produced these observations and thus the state sequence is "hidden".

The formal definition of a HMM is as follows:

$$\lambda = (A, B, \Pi) \tag{4.14}$$

s is our state alphabet set, and **v** is the observation alphabet set:

$$\mathbf{s} = (s_1, s_2, \ldots, s_N) \tag{4.15}$$

$$\mathbf{v} = (v_1, v_2, \ldots, v_M) \tag{4.16}$$

We define **q** to be a fixed state sequence of length T, and corresponding observations **o**:

$$\mathbf{q} = (q_1, q_2, \ldots, q_T) \tag{4.17}$$

$$\mathbf{o} = (o_1, o_2, \ldots, o_T) \tag{4.18}$$

A is a transition array, storing the probability of state j following state i. Note the state transition probabilities are independent of time:

$$A = [a_{ij}], \quad a_{ij} = P(q_t = s_j | q_{t-1} = s_i) \tag{4.19}$$

B is the observation array, storing the probability of observation k being produced from the state j, independent of t:

$$B = [b_i(k)], \quad b_i(k) = P(x_t = v_k | q_t = s_i) \tag{4.20}$$

Π is the initial probability array:

$$\Pi = [\pi_i], \quad \pi_i = P(q_1 = s_i) \tag{4.21}$$

Two assumptions are made by the model. The first, called the Markov assumption, states that the current state is dependent only on the previous state, this represents the memory of the model:

$$P(q_t | q_{1:t-1}) = P(q_t | q_{t-1}) \tag{4.22}$$

The independence assumption states that the output observation at time t is dependent only on the current state, it is independent of previous observations and states:

$$P(o_t | o_{t-1}, q_t) = P(o_t | q_t) \tag{4.23}$$

Given a HMM λ, and a sequence of observations **o**, the probability of the observation sequence given a model $P(\mathbf{o}|\lambda)$, can be evaluated. This problem could be viewed as one of evaluating how well a model predicts a given observation sequence, and thus the most appropriate model from a set can be chosen. In order to avoid the redundant calculations in directly evaluating $P(\mathbf{o}|\lambda)$(refer to [101] for details), the forward probability variable is introduced.

$$\alpha_t(i) = P(o_1, o_2, \ldots, o_t, q_t = s_i | \lambda) \tag{4.24}$$

The algorithm for calculating $P(\mathbf{o}|\lambda)$ is called the forward algorithm and is as follows:

1. Initialization:
$$\alpha_1(i) = \pi_i b_i(o_1), \quad 1 \leq i \leq N. \tag{4.25}$$

2. Induction:
$$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(i) a_{ij}] b_j(o_{t+1}), \quad 1 \leq t \leq T-1, 1 \leq j \leq N. \tag{4.26}$$

3. Termination:

$$P(\mathbf{o}|\lambda) = \sum_{i=1}^{N} \alpha_T(i). \tag{4.27}$$

Besides the evaluation of the observation given HMMs, there are two other key issues related to the HMM: decoding and learning. The aim of decoding is to discover the hidden state sequence that was most likely to have produced a given observation sequence. One solution to this problem is to use the Viterbi algorithm to find the single best state sequence for an observation sequence. The aim of learning is to estimate the model parameters $\lambda = (A, B, \Pi)$ that best describe a process given a set of examples from this process. For this purpose, the Baum-Welch algorithm is widely used. To save the space in the text, please refer to [101] for the details of these two algorithms.

The validity and performance of a HMM-based approach to the classification of isolated image sequences have been demonstrated in the past. However, many applications in a natural environment – like our gesture recognition task – require an automatic temporal segmentation. This problem is related to the procedure of keyword spotting in speech recognition where HMMs are successfully used for a long time. In keyword spotting, each reference pattern is defined by a keyword model and all the other patterns are modeled by a single HMM called a garbage model or a filler model. The garbage model in speech recognition represents acoustic nonkeyword patterns. It is usually trained using a finite set of nonkeyword samples. But there is a main difference: continuous speech is composed of a defined and countable number of keywords and non-keywords, whereas in the case of a continuous video stream a defined number of key image sequences is embedded in a background of an indefinite number of movements and transitions.

Threshold Model: For correct gesture spotting, the likelihood of a gesture model for a given pattern should be distinct enough. Unfortunately, although the HMM recognizer chooses a model with the best likelihood, we cannot guarantee that the pattern is really similar to the reference gesture unless the likelihood value is high enough.

A simple thresholding of the likelihood often does not work. Therefore, Lee [65] proposed a new concept, called threshold model, that yields the likelihood value to be used as a threshold. A gesture is recognized only if the likelihood of the best gesture model is higher than that of the threshold model. It is an ergodic model with the states copied from all gesture models in the system and then fully connect the states (see Figure 4.1). In this model, each state can be reached by all other states in a single transition. Output observation probabilities and self-transition probabilities in this model are kept as in the gesture models, but all outgoing transition probabilities are equally assigned as

$$a_{ij} = \frac{1 - a_{ij}}{N - 1}, \quad \text{for all } j, i \neq j \tag{4.28}$$

where a_{ij} is the transition probability from state s_i to s_j and N is the number of states (the sum of all states excluding the start and final states). The start and final states produce no observation.

The new ergodic model can match any described gestures. However, a gesture is better described by the dedicated model because the temporal order of subpatterns is better

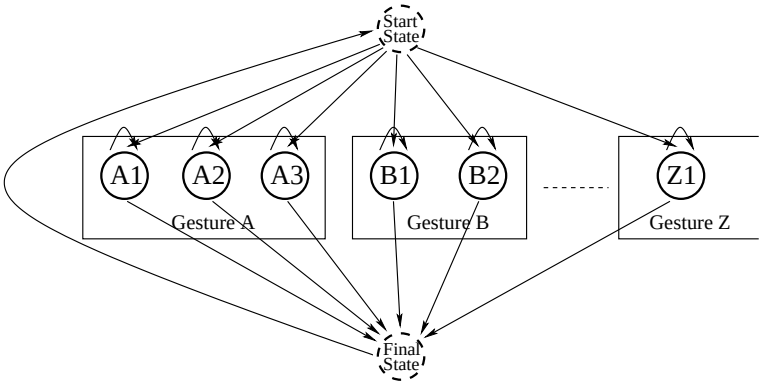


Figure 4.1: The threshold model introduced by Lee and Kim [65]

described in the model. These characteristics illustrate that the likelihood of the new model can be used for an adaptive threshold of the likelihood of gesture models. In this sense, the new ergodic model is termed threshold model. The threshold model differs from the garbage model in that its likelihood provides a confidence limit for the likelihoods calculated by other gesture models, while that of the garbage model is just a similarity measurement.

As described by Lee and Kim [65], the likelihood of the target gesture model soars up above the threshold when the forward pass gets close to the end of a gesture. The time satisfying such a condition can be called a candidate end point (CEP). Once we obtain CEP, its corresponding start point can easily be found by backtracking the Viterbi path to the start state of left-right HMM. Several constraints are set for choosing the optimal CEP. One limitation of the model is that the threshold model is constructed by combining all the gesture models in the system, the number of states in the threshold model is equal to the sum of the states of all gesture models excluding the start and final states. This means that the number of states in the threshold model increases as the number of gesture models increases.

Normalized Viterbi Algorithm: The normalized Viterbi Algorithm is introduced by Morguet and Lang [81]. The basic idea is to use sliding windows and form a local evaluation of the observations. Using the logarithm of the state probability density function (*pdf*) $\bar{f}_{s_i,t} = log(f_{s_i}(o_t))$ and the transition probabilities $\bar{a}_{i,j} = log(a_{i,j})$, the Viterbi algorithm recursively accumulates and maximizes the local score $D_{s_i,t}$ for every HMM state:

$$D_{s_i,t} = \max_j [D_{s_j,t-1} + \bar{a}_{i,j}] + \bar{f}_{s_i,t}. \tag{4.29}$$

The output score, which is the score $D_{s_N,t}$ of the last state, is crucial to the continuous recognition process. But the standard Viterbi algorithm cannot be used since, depending on the average state pdfs $\bar{f}_{s_i,t}$, the output score will permanently increase or decrease on the average. To stabilize the average score, it has to be normalized to its respective Viterbi path length.

For that reason, a local path length $L_{s_i,t}$, which allows recombining paths to have different lengths, is introduced

$$D_{s_i,t} = \max_j \left[\frac{D_{s_j,t-1} \cdot L_{s_j,t-1} + \bar{a}_{i,j} + \bar{f}_{s_i,t}}{L_{s_j,t-1} + 1} \right]. \qquad (4.30)$$
$$L_{s_i,t} = L_{s_k,t-1} + 1 \quad \text{with } k = \text{index of best } s_j$$

The output scores of an HMM λ_i are smoothed by averaging the scores in an interval between $-\tau$ and $+\tau$. The end of gestures can be detected by finding the peaks which are greater than a model dependent rejection threshold $D_{th}^{\lambda_i}$. The model dependent threshold is expressed by a single relative rejection threshold S_{rel} with the help of the model specific maximum and minimum scores:

$$D_{th}^{\lambda_i} = D_{\max}^{\lambda_i} - S_{rel} \cdot [D_{max}^{\lambda_i} - D_{min}^{\lambda_i}] \qquad (4.31)$$

In the experiment, this method achieved good results on detecting a set of predefined gestures. By tuning the value of S_{rel}, the system can achieve a balance between the recognition rate (ratio of correctly recognized gestures to the total number of key gestures) and false acceptance rate (ratio of the number of wrongly accepted gestures to the number of key gestures). One point which is not clear is how the model specific maximum and minimum scores are chosen.

4.3 Manipulative Primitive Modeling and Detection

In the previous section, the basic techniques – DTW and HMM for trajectory matching and their extensions for elementary gesture detection in continuous observation are introduced. Although both of them give a similarity between any two trajectories and the matching decision is made by choosing an appropriate threshold, in practice, the DTW algorithm is more assumed to match the trajectories which are experiencing speed difference or shape distortion, the HMMs are more suitable for the signals which have similar state propagation and a Gaussian random error of observation. If the signal does not comply with the assumptions, their classificatory capability will be inhibited. Therefore, in this section, a detailed analysis of our observation will be presented. According to the properties of the signals, the manipulative primitive models are constructed. To match the observations using these models, the sequential Monto Carlo method is chosen by us. Its basic theory will be described in the second part of this section. The concrete algorithm dedicated to our matching purpose will be put forward in detail in the third part of the section. It is named particle filter realized hidden Markov model matching (PFHMM).

4.3.1 Primitive Model

The recognition scenario for our system has been discussed in Section 2.2.2. The robot observes and aims to detect the manipulative actions, which happen on a table top, from different view-angles. The approach chosen by us was briefed in Section 3.1.2. Different to the approaches either using 3-D gesture models or being based on 2-D view-invariant features, our approach is focusing on modeling the variance of the trajectories in an appropriate way.

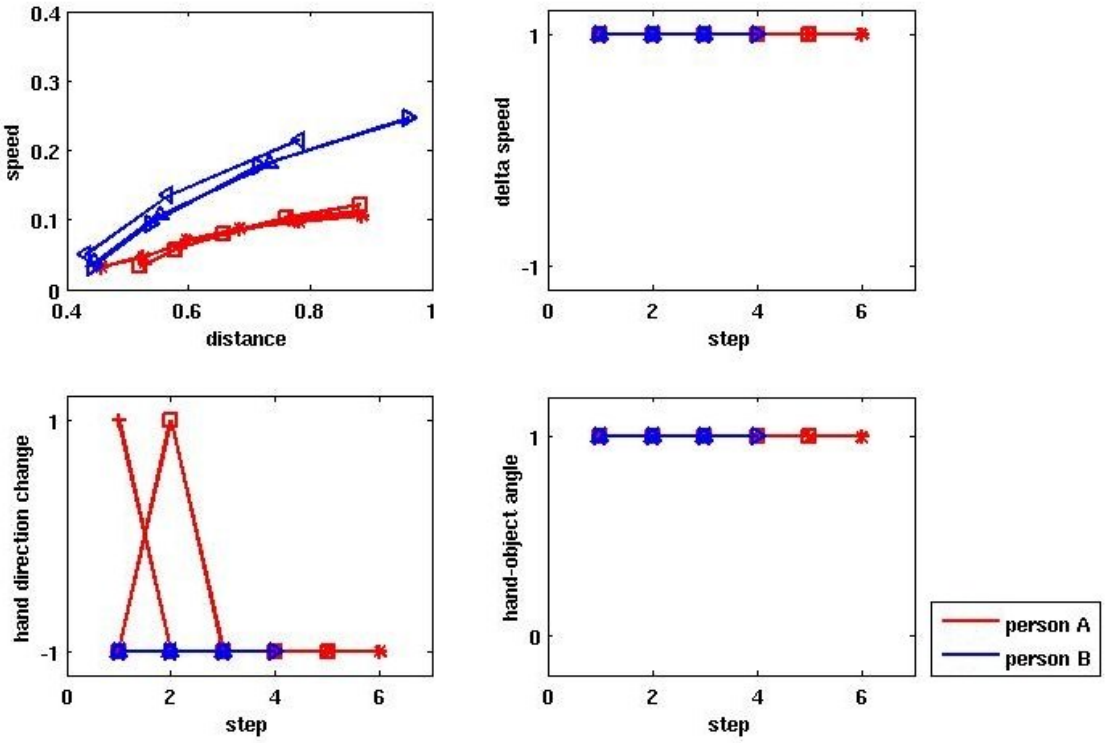


Figure 4.2: The trajectories of manipulative primitives: comparison between different persons and repeats

In order to achieve a representation of the motions of manipulative actions, a semi-continuous feature vector is constructed (see Chapter 3) which is aimed to achieve the smaller inner-class variance and larger inter-class differences. The acquired trajectories are shown in Section 3.3.3.

Here, it is necessary to have a look at the trajectories from different aspects. Figure 4.2 shows several repeats of primitive "putdown cup" performed by different persons (using different colors). The actions were filmed by the same camera. Figure 4.3 shows several repeats of primitive "putdown cup" performed by the same person but observed from different view angles (different view-angles are indicated by different colors). The top-left subfigures of both figures are presenting the trajectories of continuous features in speed-distance coordinate. The discrete features are displayed in other subfigures in feature-time coordinate. Because the discrete features are coarse discretized and are quite similar within both figures, we pay more attention to these two top-left subfigures. These subfigures show three kinds of variations of the trajectories: the variance of the repetition of a primitive by one performer, the variance of the performance of a primitive from different performers, and the variance of the observation on a primitive performed by a person but from different view-angles. It can be found that in both subfigures, the variance of the repeats of a primitive by one performer is smaller than the other two. The variance caused by different performers and the variance caused by different view-angles are comparable. If the system is proposed to achieve a person independent recognition, the variance of

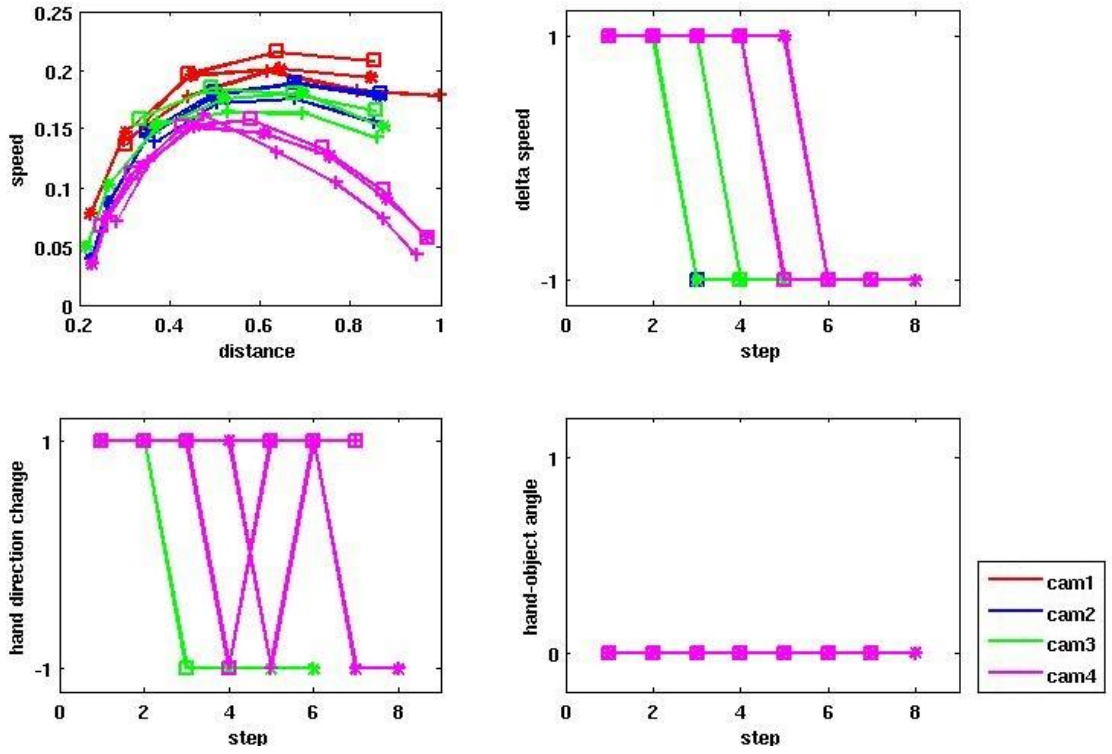


Figure 4.3: The trajectories of manipulative primitives: comparison between different views and repeats

a primitive is the accumulation of all three kinds of variances. The problem is that the model with such huge tolerance to the variance will cause many insertion errors during the continuous primitive detection. But for a personal robot, a person dependent recognition is also reasonable. Therefore, only the variance of repetition and the variance caused by different view-angles are taken into consideration.

When we have a close look of the top-left subfigure of Figure 4.3, it can be found that the variance of repetition keeps stable even when the repeats are observed from different view-angles. The effect of a different view-angle on a set of trajectories is not dispersing the set of trajectories but a little shifting and distortion of the complete set. It is like a systematic error which causes bias in measurement which leads to measured values being systematically too high or too low. What is worth notice is that the "bias", which comes from the view variant observation is not linear to the original model. It is different to the parametric HMM proposed by Wilson [127], in which a linear scaling is supposed.

Therefore, in our system the two different variances are handled differently. For the basic model, the HMM is used by us because it can handle both the spatial and temporal variance of the trajectories and has a complete learning strategy which is missed by template matching techniques like DTW. Then, the repetition variance of the primitives is covered by the observational probabilities of the HMMs. For the variance caused by different view-angles, an extra node is added into the topology of the HMM, which is used to tune the mean value of the HMMs. The model can be represented by a dynamic Bayesian

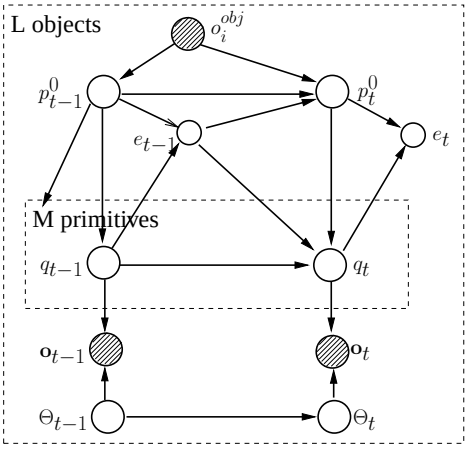


Figure 4.4: The manipulative primitive model shown as dynamic Bayesian Network

Network (see Figure 4.4). It is a two-time-slice representation. The directed arrows indicate the dependencies between the states in the current and between the consecutive time slices. Because of the object-oriented approach (see Section 3.3.1), the possible HMMs p_t^0 are generated according to the object detection o_t^{obj}. The HMMs p_t^0 have corresponding hidden states q_t. Different to the normal parameter set $\lambda = (A, B, \Pi)$ of an HMM, a terminal probability E is added. It reflects the terminal probability of an HMM given a hidden state q. Therefore, based on the current HMM p_t^0 and hidden state q_t, the state of node e_t can be estimated on every time step . It is used to indicate whether the HMM ends here and to influence the prediction of the nodes in the next time slice. o_t is the observation of the hidden state q_t. The observational probability here covers the variance of primitive repetition. The node θ_t is an extra parameter scaling the observation. It has an initial value and changes slightly over times. With this property it can tune a HMM to fit the observation from different view-angles.

However, the scaling parameter can not be estimated before the observations are available. The traditional algorithms to evaluate the matching between the observation and HMMs like forward algorithm can not be used here. Therefore, the particle filter widely used in the recursive Bayesian filtering is chosen by us. It is a technique for implementing the recursive Bayesian filter by Monte Carlo simulations. In the next section, the Sequential Monte Carlo Method will be introduced. After that, the algorithm particle filter realized Hidden Markov Model Matching will be described in detail.

4.3.2 Sequential Monte Carlo Method for Trajectory Matching

Let us assume that the human activity is given by a sequence $\mathbf{o}_t$.

$$\mathbf{o}_t = (o_1, \ldots, o_t) \tag{4.32}$$

For each time step t, the state of the stochastic process giving rise to the observation o_t is denoted by the random variable q_t. To simplify the representation, q_t and o_t are written as scalar variables, but in general they could be vectors. As a statistical model, the aim

of our analysis is to compute the distribution $P(q_t|\mathbf{o}_t)$. According to the Bayes's rule, it turns to:

$$\begin{aligned} P(q_t|\mathbf{o}_t) = P(q_t|o_t, \mathbf{o}_{t-1}) &= \frac{P(o_t|q_t, \mathbf{o}_{t-1})\ P(q_t|\mathbf{o}_{t-1})}{P(o_t|\mathbf{o}_{t-1})} \\ &= c\ P(o_t|q_t, \mathbf{o}_{t-1})\ P(q_t|\mathbf{o}_{t-1}) \\ &= c\ P(o_t|q_t)\ P(q_t|\mathbf{o}_{t-1}) \end{aligned} \tag{4.33}$$

In this equation, $P(q_t|\mathbf{o}_{t-1})$ is the *a priori* density and $P(q_t|\mathbf{o}_t)$ is the *a posteriori* density respectively. It is based on whether before or after the measurement at time t is incorporated. $c = 1/P(o_t|\mathbf{o}_{t-1})$ is a normalization factor independent of q_t. According to the markov process assumption, the conditional observation density $P(o_t|q_t)$ for a measurement at time t is independent of the observation history $\mathbf{o}_{t-1}$, and depends only on the current state pdf:

$$P(o_t|q_t) = P(o_t|q_t, \mathbf{o}_{t-1}) \tag{4.34}$$

The $P(q_t|\mathbf{o}_{t-1})$ is the prior from the accumulated observation history up to time $t-1$. This is equivalent to the posterior at the previous time step $P(q_{t-1}|\mathbf{o}_{t-1})$ predicted to the actual time step:

$$P(q_t|\mathbf{o}_{t-1}) = \int P(q_t|q_{t-1})P(q_{t-1}|\mathbf{o}_{t-1})\,\mathrm{d}q_{t-1} \tag{4.35}$$

Based on this model, a human activity can be tracked over time by calculating at every time step first the *a priori* density $P(q_t|\mathbf{o}_{t-1})$ from Eq. 4.35 and then evaluating the *a posteriori* density $P(q_t|\mathbf{o}_t)$ with Eq. 4.33 based on the new measurement o_t. This method to track the state probability density function over time with integration of measurements is known as recursive Bayesian filter.

Different to Kalman filter and HMM, which are also recursive Bayesian filters, sequential Monte Carlo (SMC) method could model a non-linear, non-Gaussian state *pdf* by approximating it using a large set of weighted samples, named particles. When it is used for filtering purpose, it is also called particle filter. These particles are propagated over time using simple importance sampling (IS) and resampling mechanisms. Asymptotically, i.e. as the number of particles goes to infinity, the convergence of these particle approximations towards the sequence of probability distributions can be ensured under very weak assumptions.

The complete set of particles is written as:

$$\left\{(s_t^{(1)}, \pi_t^{(1)}), \ldots, (s_t^{(N)}, \pi_t^{(N)})\right\}, \tag{4.36}$$

where, N is the number of particles, each sample $s_t^{(i)}$ has its associated weight $\pi_t^{(i)}$.

For $N \to \infty$, the overall *a posteriori* probability $P(\mathbf{q}_t|\mathbf{o}_t)$ at time t given the sequence of states $\mathbf{q}_t = \{q_1, q_2, \ldots, q_t\}$ and samples $\mathbf{s}_t^{(i)} = \{s_1^{(i)}, s_2^{(i)}, \ldots, s_t^{(i)}\}$ can then be represented by

$$P(\mathbf{q}_t|\mathbf{o}_t) \approx \sum_{i=1}^{N} \pi_t^{(i)} \delta(\mathbf{q}_t - \mathbf{s}_t^{(i)}) \tag{4.37}$$

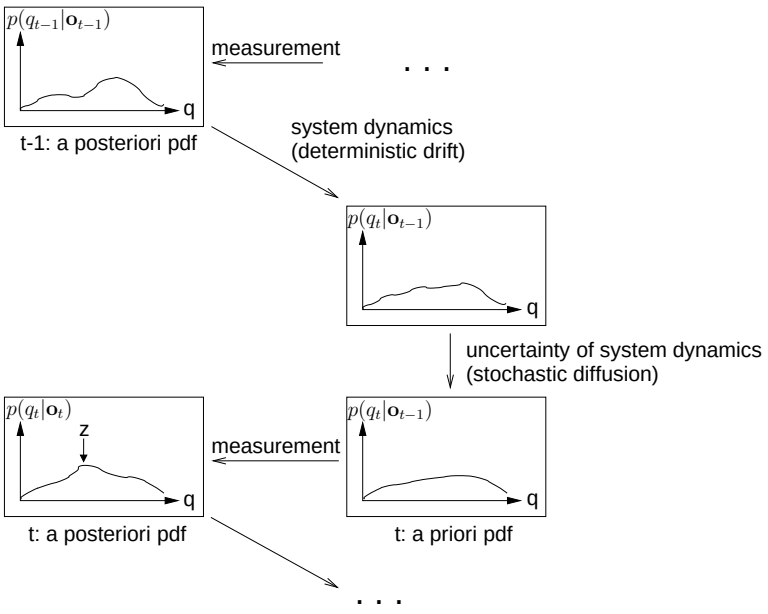


Figure 4.5: The propagation of the sample set in a particle filter (from [33])

The sequence of observations $\mathbf{o}_t = \{o_1, o_2, \ldots, o_t\}$ is implicitly contained in Eq. 4.37 as for every time step t the observation o_t is used for calculating the weights $\pi_t^{(i)}$. The temporal propagation of the N weighted samples is carried out through first propagating the samples to the new time step $t+1$ and then updating the weights. The state propagation usually contains a deterministic drift based on the system dynamics and some diffusion resulting from the uncertainty in the system dynamics (see Figure 4.5).

Sequential Importance Sampling The original method used for propagating a sample distribution over time is sequential importance sampling (SIS). Assume that there is a probability density $f(y)$ that is difficult to sample because it is what we want to estimate but for which a proportional pdf $r(y)$ is available that can be evaluated:

$$f(y) \propto r(y) \tag{4.38}$$

If we have an *importance density* $t(y)$ from which we can easily draw samples $x^{(i)}$, we can calculate a weighted approximation to $f(y)$ by:

$$f(y) \approx \sum_{i=1}^{N} \pi^{(i)} \delta(y - x^{(i)}) \quad with \quad \pi^{(i)} \propto \frac{r(x^{(i)})}{t(x^{(i)})} \tag{4.39}$$

In this approximation, the weights $\pi^{(i)}$ compensate for the difference between the importance density $t(y)$ and the probability density $f(y)$.

Using importance sampling, the weights in Eq. 4.37 can be calculated. This requires an importance density $t(\mathbf{q}_t|\mathbf{o}_t)$ for drawing samples $\mathbf{s}_t^{(i)}$ giving:

$$\pi_t^{(i)} \propto \frac{P(\mathbf{s}_t^{(i)}|\mathbf{o}_t)}{t(\mathbf{s}_t^{(i)}|\mathbf{o}_t)} \tag{4.40}$$

If only a filtered estimate $P(q_t|\mathbf{o}_t)$ is needed at sequential time steps and the human activity can be modeled by a first-order Markov process, it can be shown (see [2]) that the weights can be calculated sequentially from:

$$\pi_t^{(i)} \propto \pi_{t-1}^{(i)} \frac{P(o_t|s_t^{(i)})P(s_t^{(i)}|s_{t-1}^{(i)})}{t(s_t^{(i)}|s_{t-1}^{(i)}, o_t)} \tag{4.41}$$

After a few iterations of the weight calculation using this type of importance sampling, a few samples will have high weights while most of the samples will have negligible weights. This degeneracy phenomenon cannot be avoided in the standard SIS approach and results in a waste of computational power for updating a large amount of particles having small weights. To circumvent the degeneracy phenomenon, either the importance density must be chosen very carefully or a resampling of the sample distribution must be introduced. Today there exists a wide range of particle filtering approaches that follow the basic SIS algorithm outlined above but differ in the choice of importance density and/or resampling step.

Sampling Importance Resampling A specific particle filtering technique called sampling importance resampling (SIR) is better known as *Conditional Density Propagation* – CONDENSATION introduced by Isard and Blake [52] to track objects in noisy image sequences. In the SIR algorithm the importance density is chosen to be the prior density:

$$t(s_t^{(i)}|s_{t-1}^{(i)}, o_t) = P(s_t^{(i)}|s_{t-1}^{(i)}) \tag{4.42}$$

This leads to a simplification of the weight update Eq. 4.41:

$$\pi_t^{(i)} \propto \pi_{t-1}^{(i)} P(o_t|s_t^{(i)}) \tag{4.43}$$

Resampling is performed in SIR at every time step, so that $\pi_{t-1}^{(i)} = 1/N$ and the weight update becomes:

$$\pi_t^{(i)} \propto P(o_t|s_t^{(i)}) \tag{4.44}$$

Notice that the importance density is independent of the observation o_t and, consequently, the state space is explored without any knowledge of the observation. Additionally, resampling in every time step could result in a loss of diversity among the samples, as the samples with high weights will be selected very often during resampling. If the process noise is small, the samples selected several times during resampling will not differ very much after the sample propagation. Therefore, the sample set will eventually contain multiple instances of the same sample. However, in computer vision applications the observations obtained from image data are usually very noisy so that sample set degeneracy is very unlikely.

Following the original publication, this tracking framework has been extended to automatically switch between several movement models to provide a mechanism for classification of the movements [53]. For this purpose, a multinomial label μ is added to each sample $\mathbf{s}_t^{(i)}$ indicating the movement model the sample belongs to:

$$\mathbf{s}^{(i)} = (\mathbf{x}, \mu) \quad with \quad \mathbf{x} = (x_1, \ldots, x_m), \quad \mu \in \{1 \ldots l\} \tag{4.45}$$

A model specific sample propagation from time t to time $t+1$ is used to propagate the samples depending on the model they represent. The recognition of a specific model is realized by calculating at every time step t and for every model μ the *model probability* $P_t(\mu = j)$ based on all samples belonging to this model:

$$P_t(j) = \sum_{i \in \Upsilon_j} \pi_t^{(i)} \quad with\ \Upsilon_j = \{k \mid \mathbf{s}_t^{(k)} = (\mathbf{x}, \mu = j)\} \tag{4.46}$$

Now at every time step the highest model probability indicates which model is currently dominant in the state space and therefore best represents the observed data.

4.3.3 Particle Filter Realized Hidden Markov Model Matching

As already discussed in the Section 4.3.1, the features of manipulative primitives vary with different viewpoints. In contrast to the variance introduced by different persons or by different performances of the same person, it is a kind of systematic error that we aim to compensate for. Given that we have no apriori information on the current view angle, we model this influence by an additional hidden variable which is adapting the mean value of the observation probability (see Figure 4.4). As a consequence, the robot only needs to observe the action from one point of view during learning and afterwards recognize it from a significantly different view-angle in a certain range.

The underlying PF is using SIR. The matching of the HMM and the observation are achieved by temporal propagation of a set of weighted particles:

$$\{(\mathbf{s}_t^{(1)}, w_t^{(1)}), \ldots, (\mathbf{s}_t^{(N)}, w_t^{(N)})\} \tag{4.47}$$

with

$$\mathbf{s}_t^{(i)} = \{p_t^{0(i)}, q_t^{(i)}, e_t^{(i)}, \Theta_t^{(i)}\} \tag{4.48}$$

The number of particles is N. The sample $\mathbf{s}_t^{(i)}$ contains the primitive index $p_t^{0(i)}$, the hidden state $q_t^{(i)}$, the terminal state of this primitive $e_t^{(i)}$ at time t, and the observation scaling vector $\Theta_t^{(i)}$. The weight $w_t^{(i)}$ of a sample can be calculated from

$$w_t^{(i)} = \frac{P(\mathbf{o}_t|\mathbf{s}_t^{(i)})}{\sum_{j=1}^{N} P(\mathbf{o}_t|\mathbf{s}_t^{(j)})}. \tag{4.49}$$

Here, $P(\mathbf{o}_t|\mathbf{s}_t^{(i)})$ models the observation probability of the scaled $\mathbf{o}_t$ given $q_t^{(i)}$ and HMM $p_t^{0(i)}$. Let $o_{t,m}$ be the m^{th} component of the observation vector at time t, then, $P\{o_{t,m}|\mathbf{s}_t^{(i)}\}$ is calculated as

$$P\{o_{t,m}|\mathbf{s}_t^{(i)}\} = \frac{1}{\sqrt{2\pi}\sigma_{\mathbf{s}_t^{(i)},m}} \texttt{exp} \frac{-(o_{t,m} - \theta_{t,m} \cdot \mu_{\mathbf{s}_t^{(i)},m})^2}{2\sigma^2_{\mathbf{s}_t^{(i)},m}} \tag{4.50}$$

The $\mu_{\mathbf{s}_t^{(i)},m}$, $\sigma_{\mathbf{s}_t^{(i)},m}$ is the mean and standard deviation of the m^{th} component given the hidden state $\mathbf{s}_t^{(i)}$, which is determined by $q_t^{(i)}$ and $p_t^{0(i)}$. The effect of the scaling parameter $\theta_{t,m}$ is shown in Figure 4.6. Suppose the red line is the model and the green line is the

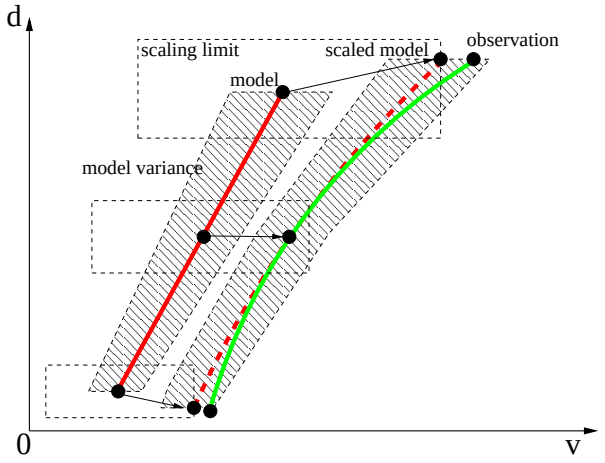


Figure 4.6: The effect of scaling parameter

observation, the model can be scaled to the dashed red line according to the allowed scaling limits (the dashed rectangles), where they achieve a matching. Note that The original model variance keeps unchanged (the two shadowed areas). With the survival of the fittest rule, the PF will choose the best value within a limited range of the scaling parameters through the particle propagation.

The propagation of the weighted samples over time consists of three steps:
Select: Selection of $N - M$ samples $\mathbf{s}_{t-1}^{(i)}$ according to their respective weight $w_{t-1}^{(i)}$ and random initialization of M new samples. In every newly initialized sample, the value of $\theta_{t,j}^{(i)}$ is randomly chosen from its allowed period.
Predict: The current state of each sample $\mathbf{s}_t^{(i)}$ is predicted from the samples from the select step according to the graphical model given in Fig. 4.4. The terminal state $e_{t-1}^{(i)}$ is a bi-valued variable, 0 means the primitive is continuing and 1 means the primitive ends here. So if $e_{t-1}^{(i)}$ is 0, the next hidden state $q_t^{(i)}$ is sampled according to the transition probability of the HMM of primitive $q_{t-1}^{(i)}$ and the primitive index $p_t^{0(i)}$ keeps the same as $p_{t-1}^{0(i)}$. The $\theta_{t,j}^{(i)}$ is updated by $\theta_{t-1,j}^{(i)} + \mathcal{N}(\sigma_{\theta_j})$. $\mathcal{N}(\sigma_{\theta_j})$ is a normal distribution and represents the uncertainty in the prediction of θ_j, which spans a much smaller area than the allowed period. If the terminal state $e_{t-1}^{(i)}$ is 1, the primitive index $p_t^{0(i)}$ will be sampled according to the current possible primitives of this object. Then the hidden state $q_t^{(i)}$ is sampled according to the initial probability of the HMM of the new primitive $p_t^{0(i)}$. The $\Theta^{(i)}$ is reinitialized. At the end of this step, the terminal state of this particle $e_t^{(i)}$ is sampled based on the terminal probability of the current primitive state $q_t^{(i)}$.
Update: Determination of the weights $w_t^{(i)}$ of the predicted samples $\mathbf{s}_t^{(i)}$ using Eq. 4.49.

The recognition of a manipulative primitive is achieved by calculating the *end-probability* P_{end} that a certain HMM model p_i is completed at time t:

$$P_{\text{end},t}(p_i) = \sum_n w_t^{(n)}, \text{ if } p_i \in \mathbf{s}_t^{(n)}. \tag{4.51}$$

A primitive model is considered recognized if the probability $P_{\text{end},t}(p_k)$ of the primitive model p_k exceeds a threshold p^0_{th} which has been determined empirically.

The resampling step in the particle propagation is able to adapt the starting point of the model matching process if the beginning of the primitive does not match the beginning of the segment. The *end-probability* gives an estimation of the primitive's ending point. This combination to a certain extent solves the problem of the forward-backward algorithm which needs a clear segmentation of the pattern.

4.4 Evaluation

In our experiment, a scenario in an office environment is set up as shown in the images in Figure 3.6 on page 35. A person is sitting behind a table and manipulates the objects that are located on it. She or he is assumed to perform one of three different manipulation tasks: (1) *water plant*: take cup, water plant, put cup; (2) *prepare tea*: consists of take/put cup, take tea can/sugar, pour tea/take sugar into cup, put tea can; (3) *prepare coffee*: consists of take/put cup, take milk/sugar, pour milk/take sugar into cup, put milk. The images are recorded by 4 cameras at the same time with a resolution of 320x240 pixels and with a frame-rate of 15 images per second. The positions and view angles of the cameras have been shown in Table 3.1 in the previous chapter. The distance to table and the pitch angle of the camera are illustrated in Figure 3.3. The horizontal angles of the cameras are indicated in Figure 3.6. They have the same height of 160cm as the robot BIRON [44]. In the experiment, each task is performed 15 times with different object layouts by 2 different persons. The object recognition results have been labeled because the evaluation experiment should concentrate on the performance of the trajectory matching.

To test the performance of the scaling parameter in the primitive detection, the training data for each primitive are taken from the same single camera (camera 2). Furthermore, each training sample of a specific action was performed with regard to an object at same place on the table. In this study, the primitive training is person-specific. Then, the learned models are applied to the data taken from all 4 cameras. Table 4.1 compares the detection results of the primitives related to the objects between the methods with and without scaling. The results are calculated based on the primitives from all camera views with parameter set $N = 1000$, $M = 50$, $p^0_{th} = 0.2$, $\beta = 3$, the scaling range [0.8 1.2] with predict variance $\sigma = 0.1$. The primitives are modeled by semi-continuous HMMs with left-right topology. The parameters are learned from manually segmented trajectories with the Baum-Welch algorithm, E is calculated similar to Π, except using the last states. To present the quality of the detection because the primitives are detected from long trajectories, the primitive error rate (PER) is defined as

$$\text{PER} = \frac{\#\text{Substitution} + \#\text{Insertion} + \#\text{Deletion}}{\#\text{Truth}} \tag{4.52}$$

The ground truth of every primitive has a time stamp and an allowed time variance for matching. No detection in the allowed time variance of a primitive causes a deletion error. A false detection in it is counted as a substitution. An insertion error is a detection out of that range or an additional detection in the range. Table 4.1 shows the results. It

Table 4.1: The detection of the manipulative primitives.

Object	Primitive	Num. of Truth	PER(%)	PER(%, no scale)
tea	take	15	41.6	39.5
	put	15	8.3	19.8
milk	take	12	14.4	14.4
	put	12	42.1	52.6
sugar	take	15	6.3	16.9
cup	take	45	17.8	27.1
	put	45	11.6	22.2
	pour	42	32.7	36.9
plant	water	15	20.8	19.8

can be found that generally the PER of the method with scaling parameter is lower than that without it for most primitives. But there are also few cases where it increased. In the experiment, the big milk carton is placed near the tea can (see Figure 3.6 Cam3), which caused much insertion error in the PERs of "take tea" and "put milk". In oder to investigate the results in detail, we split the PER into different error types. Figure 4.7 shows the error rates of different types by using both methods. The numbers summarized all the results of all primitives and from all cameras. According to this figure, the scaling caused a significant drop of the error rates caused by deletion, as we expected. It also bring a slight increase of the insertion errors because it generalized the HMMs.

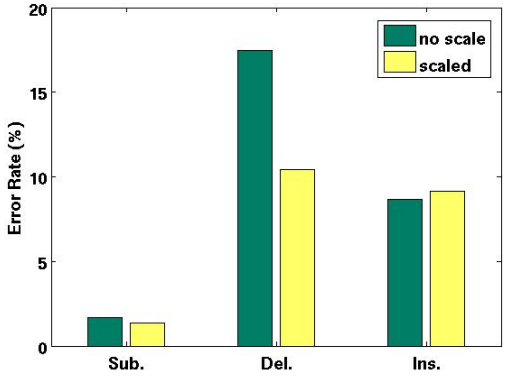


Figure 4.7: The primitive detection results shown by different error types and with/without scaling

Figure 4.8 presents the effect of the scaling parameter on the results for different cameras. Note that the observation from one camera could contain the manipulations on different object layouts, the results for camera 2 is also listed here. The PERs for all cameras are decreased by using it. Especially, the result with regard to the camera 4, which has the largest PER because of the largest view-angle difference, achieved bigger improvement than the others.

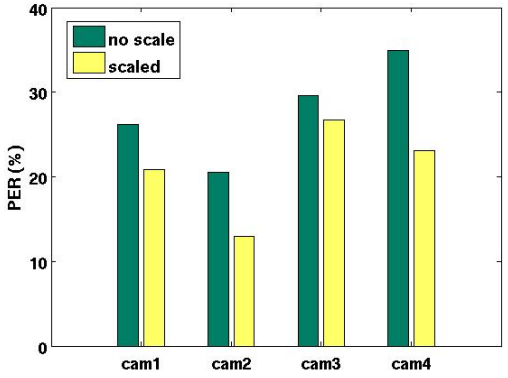


Figure 4.8: The primitive detection results with regard to different camera views

4.5 Summary

In the last chapter, we constructed a feature vector for the manipulative gestures. It is a 2-D representation but less affected by the change of view-angles. The object vicinity was also put forward to pre-segment the hand trajectory. In this chapter, we are focusing on how to detect the manipulative primitives from the pre-segmented trajectories. The difficulty lies on two points. Firstly, the pre-segmented trajectories could also contain meaningless parts. This requires that the primitive can be spotted out of longer observations. Secondly, the primitives observed from different view-angles have much more variance. After a close look of different kinds of variances, a person-specific manipulative gesture recognition is chosen by us. HMMs are used by us to model the primitives. In addition, we added end probabilities to have an estimate of the end points of the primitives. Then, a particle filter realized HMM matching method is introduced. It used weighted particles to track the primitive models which have better matching to the observations. With an online evaluation of the end probability of a primitive, the end points of the primitives also can be found during a continuous observation. In this process, a dynamic scaling parameter in the observation model is used to cope with the nonlinear changes of the trajectories caused by different view-angles. The experiments showed that the manipulative primitives can be robustly detected from the trajectories and the scaling parameter promoted these results.

5. Manipulative Task Modeling and Recognition

In the previous chapter, the model for the manipulative primitive was presented. However, it is a flat recognition model which can not handle the hierarchical structure within a sequence of manipulations that have a more complex overall meaning. This is a limitation which severely constrains the robot's understanding competence. For example, the sequential manipulation of "take a pencil" and "move it to a notebook" should not only be recognized as two independent manipulative actions but also as an entity with the underlying human intention "to write". For a personal robot, it is necessary to recognize human manipulative activities and the associated intentions that are often inferred by human observers. It has been shown by the cognitive developmentalists that even one-year-old infants can infer intentions before the whole activity is completed [41].

Although the concept of the motion primitive has been widely accepted as a basic unit to generate a smooth performance of an action, it is fair to say that there is no persuasive research results showing the general structuring of the complex human motions, which have higher level intentions like "prepare a cup of coffee". Many researchers doing action, or namely intention recognition decompose the target actions into a set of elementary motions shared by the actions according to the domain-specific knowledge. For example, the high level maneuvers of a driver on a highway can be predicted by observing the basic driving behaviors, like "turn left", "pass drive" etc. [99]. Our approach also follows this idea. What is worth mentioning is that different to the pure symbolic approaches which start the recognition when the meaningful symbols have been labeled on the basic actions by a human, our approach tries to stay as connected to the visual signal as possible, where the particular action elements for example "take a cup" have direct visual correlates.

The bridging between the visual signal and interpretation leads to the symbol grounding problem during the modeling. One distinctive property of the gestures we want to recognize is that they are always performed on some objects. With respect to the function of an object, persons performs different actions on it (see Section 3.3.1). The object-specific

manipulative primitive are adopted by us as the basic granule for the higher level intention recognition.

As an object-centered approach, the processing is carried out in a parallel manner on the objects in the scene. Robot companions are intended for operation in private homes or offices. They are facing a cluttered environment with all kinds of objects. A lot of objects could appear in the view of a robot at the same time. This situation does not only pose much unnecessary processing load on the system but also increases the ambiguity for the intention interpretation. To avoid this problem, the use of a task-level prediction is considered by us. Based on the assumption of the possible manipulative tasks and the recognized primitives, our system restrict the relevant object types as well as the action primitives possibly recognized in the future [67].

In the following, some related work will be presented. Then, several basic models for sequential symbols are introduced. Following the technique background, the two-layer system architecture of our system and mathematic modeling of the manipulative tasks will be put forward. After that, the combined top-down and bottom-up process, which forms an attention system of the robot, is described. To prove the efficiency of the system, task recognition experiments in an office scenario are conducted. At the end, the extensibility of the structure will also be discussed.

5.1 Related Work

To help the people in everyday life, the understanding of human behaviors at different levels is very important for many potential applications. For example, monitoring and recognizing the actions of the elderly can help them in an appropriate time and extend their functional capability [39]. If the humans action is well understood, the intelligent systems like smart room [103] or service robot [66; 69] can interact with its human users in ways that are both natural and intuitive. If the movement of human doesn't happen only in a room but in a larger scale, e.g. on a floor with many rooms [90] or in a city [72], the hierarchical human activity can also be learned and used for inference or prediction purposes.

Most of the previous work on recognizing human activities was targeted at single, simple events, e.g., "waving the hand" or "pointing". The basic techniques to match sequential data were introduced in Section 4.2. A significant part of work used the extensions of HMM, e.g., parameterized HMMs [127], variable-length HMMs [38], and coupled HMMs [135] to recognized more complex activities such as the interaction between two people. They were in principal also identifying "primitives" because there is no hierarchical modeling of the event. In recent years, attempts have been made to integrate high-level behavior models with low-level sensor models. A formulation for hierarchical HMMs (HHMM) was first proposed by Fine et al. [28]. In his work, the standard Baum-Welch procedure is extended to estimate the model parameters. Because of the computational complexity of the original algorithm, Murphy and Paskin introduce a linear-time inference algorithm for HHMMs [82]. Nguyen et al. applied the HHMM with shared structures to the problem of activity recognition in a real environment and showed its superiority over tree HHMMs [86]. Rao-Blackwellised particle filter (RBPF) was adopted as the inference

algorithm. The model parameters are estimated in an integrated way, as compared with other approaches which employ level-by-level parameter estimation, e.g. the layered HMM (LHMM) proposed by Oliver et al. [90].

The LHMM approach is closely related to the concept of *stacked generalization* [90], where the main idea is to learn classifiers on the top of classifiers. Stack generalization is a technique proposed to use learning at multiple levels. A learning algorithm is used to determine how the outputs of the base classifiers should be combined. For example, in a two-layer stacked classifier, the original data set constitutes the level zero. The base classifiers are trained on the data. Another learning process occurs using the output of base classifiers as the inputs. This is a more sophisticated technique than cross-validation and has been shown to reduce the classification error due to the bias in the classifiers. HMMs are generative probabilistic models, they can be treated as classifiers at every level. Zhang et al. also used layered HMM to classify group actions in meetings based on multi-modal data [136].

Besides HMM, there are other models which can be used at higher levels, e.g., N-gram model and grammatical models. A two-level architecture is proposed by Yin, Chai and Yang to infer a user's goals in a complex indoor environment using an radio frequency based wireless network [134]. The model relies on a dynamic Bayesian network to infer a user's actions from raw signals – the trajectories, and an N-gram model to infer the users' goals from actions. Grammatical models are another technique which are often used to describe the underlying structure of a sequence of symbols. In speech recognition, the word spotting might be performed by a statistical acoustic recognizer. A subsequent stage based on a formal grammar would then provide constraints on the sequences of words. It prevents the system from generating meaningless sentences. Similar to the speech, the human activities are also performed according to certain "rules". Moore and Essa used stochastic context free grammar (SCFG) to recognize multi-tasked activities in card games [79]. Yamamoto et al. used it to recognize Japanese tea service [132]. Because the SCFG is sensitive to the symbol detection errors in the bottom layer, including insertion, substitution, and deletion, these two papers above approached the problem from different direction: on the tolerance of the grammatical structure and the precise segmentation of the actions.

In the layered approaches, the recognition of higher level intentions is dependent on the lower level signal processing. In our case, it is the detection of manipulative primitives. Because of the object-oriented approach, the relative trajectories between the hand and every detected object will be matched to the models. When there are many objects in the scene, the system is confronted with an attention problem. This problem also appears in many other researchers' work. For example, Oliver et al. proposed a combination of top-down and bottom-up information processing for outdoor visual surveillance of human interactions [91]. But in their paper, the top-down strategy have not been cleared stated. Khadhouri and Demiris used the saliency of the bottom-up part to initialize the top-down. The top-down process selects only the relevant behaviors to the demonstrated action and consequently influences the bottom-up saliency map [59]. This system is dedicated to differentiate simple actions like "pick coke can" or "pick orange". Therefore there is no primitive detection, possible behaviors are chosen based on the features like motion, color, and the size of the object. According to the predefined forward model, an action

was rewarded by one confidence point if the prediction of next state is proved by the observation. With this mechanism, the robot's attention will be guilded to an action, which gains more and more confidence over the performance. The action recognition is made by the rule "highest confidence level always wins".

5.2 Models for Symbol Sequences

Enlightened by the concept of stacked generalization, it is good to decouple the higher level activity recognition problem into several layers. A layered structure provides several valuable properties. A layered formulation makes it feasible to carry out analysis for training and inference on different levels. Once the system is trained, inference can be carried out at any level of the hierarchy. Of course, the choice and organization of the recognizers in this structure are different to each other according to the concrete problems and the form of the outputs on different layers. For example, Robertson and Reid learned distributions on low level features, e.g., trajectories and local motion, from the non-parametric training databases. Then the lower level recognizer gave the probabilities of the observations based on the learned distributions. HMMs were used on the higher level to fuse the outputs [105]. In our case, the bottom layer will output the detection results of the manipulative primitives, which are sequences of symbols. The analysis of the sequences of symbols (words) have been well investigated in the speech recognition and understanding society. The most popular models used are n-gram model and grammatical models. Recently, these models are also widely used for activity recognition. In the following, we will present these two kinds of models and present both their advantages and disadvantages.

5.2.1 N-Gram Model

The goal of statistical language modeling is to define a probability distribution over a set of symbol or word sequences. This means to calculate $P(\mathbf{w})$ for a word sequence $\mathbf{w} = w_1, w_2, \ldots, w_k$. Factorizing $P(\mathbf{w})$ using the chain rule, we get:

$$\begin{aligned} P(\mathbf{w}) &= P(w_1)P(w_2|w_1)\ldots P(w_T|w_1,\ldots,w_{T-1}) \\ &= \prod_{t=1}^{T} P(w_t|w_1,\ldots,w_{t-1}) \end{aligned} \tag{5.1}$$

The idea of the n-gram model can be traced back to an experiment by Claude Shannon in information theory. His question was, given a sequence of letters what is the likelihood of the next letter? More concisely, an n-gram model predicts w_i based on $w_{i-1}, w_{i-2}, \ldots, w_{i-n}$. In probability terms, this is nothing but $P(w_i|w_{i-1}, w_{i-2}, \ldots, w_{i-n+1})$. To restrict the length of the "history", an n-tuple is chosen instead of the entire sequence. It can be expressed as:

$$P(w_t|w_{t-1},\ldots,w_{t-n+1}) = P(w_t|w_{t-1},\ldots,w_1) \tag{5.2}$$

It is equal to a Markov chain model of order $(n-1)$. The Markov chain is used as an approximation of the true underlying language model. This assumption is important

because it massively simplifies the problem of learning the language model from data. An n-tuple of size 1 is a "uni-gram"; size 2 is a "bi-gram"; size 3 is a "tri-gram"; and size 4 or more is simply called an "n-gram". According to n-gram model, Eq. 5.1 can be simplified into:

$$P(\mathbf{w}) = \prod_{t=1}^{T} P(w_t|w_{t-n+1}, \ldots, w_{t-1}) \tag{5.3}$$

A simple approach to estimating n-gram probabilities is to use the method of maximum likelihood within each context, which results in calculating the strict relative frequencies. Specifically, given a training corpus, we can estimate the n-gram probabilities as follows:

$$P(w_t|w_{t-1}, \ldots, w_{t-n+1}) = \frac{c(w_t, \ldots, w_{t-n+1})}{c(w_{t-1}, \ldots, w_{t-n+1})} \tag{5.4}$$

where the $c(w_t, \ldots, w_{t-n+1})$ is the occurrences of strings $w_t, \ldots, w_{t-n+1}$ in the training corpus. A problem however arises if the value of the probabilities in Eq. 5.4 is zero, i.e. if an n-gram appears in the text to be recognized but is not contained in the training corpus. Such n-grams are referred as missing or non-occurring n-grams. Assigning the maximum-likelihood probability of zero to such n-grams would lead to an error when computing the log probability. Therefore, techniques named *smoothing* are developed for adjusting the maximum likelihood estimation to produce more accurate probabilities.

When we have good confidence on the training corpus, that means underlying models of the training corpus and test text have only slight difference, one solution to avoid the missing n-grams is to assign them a small probability. The simplest way is to update the n-gram probability by assuming the missing n-grams occurred exactly once in the training corpus, i.e. by assigning it the probability:

$$P(w_t|w_{t-1}, \ldots, w_{t-n+1}) = \frac{1}{c(w_{t-1}, \ldots, w_{t-n+1})} \tag{5.5}$$

At the same time, the probabilities of all other n-grams with the same n-1 preceding words should be updated due to the addition of the new n-grams. Because of the one occurrence assumption, it is also called *"add one"* method. In practice this change is very small and the updated probabilities are very close to the original ones.

There are also other smoothing methods: *discounting* and *model combination*. The idea of discounting is to reduce maximum likelihood probabilities estimated from the observed frequency counts and then re-distribute "freed" probability mass to previously unseen events. If we use $P^*()$ and $c^*()$ to represent the modified probability and the occurrence counting, the discounting of the probabilities of seen n-grams is:

$$P^*(z|\mathbf{y}) = \frac{c^*(z\mathbf{y})}{c(\mathbf{y})} = \frac{c(z\mathbf{y}) - \beta(z\mathbf{y})}{c(\mathbf{y})} \tag{5.6}$$

To simplify the equations, the n-grams are represented by $\mathbf{y}z$, $\mathbf{y}$ is the history, z is the predicted word. The non-occurrence n-gram with the history $\mathbf{y}$ has the probability $\lambda(\mathbf{y})$:

$$\lambda(\mathbf{y}) = \frac{\sum_{c(z\mathbf{y})>0} \beta(z\mathbf{y})}{c(\mathbf{y})} \tag{5.7}$$

If the discounting factor $\beta()$ is proportional to n-gram count: $\beta(\mathbf{y}z) = \alpha c(\mathbf{y}z)$, this is a linear discounting. If the discounting factor $\beta()$ is a constant, this is an absolute discounting. The "add one" method also can be thought as an instance of absolute discounting. The idea of model combination is just the naming: combine models (uni-gram, bi-gram, tri-gram, ...) in such a way that the most precise model available is used. The discount of probabilities of non-missing n-grams can be distributed to lower order n-gram models. If $\mathbf{y} = w_{t-1}, \ldots, w_{t-n+1}$, let $\hat{\mathbf{y}} = w_{t-1}, \ldots, w_{t-n+2}$ to be shortened history. The $P^*(z|\mathbf{y})$ can be achieved by "interpolation" between $P(z|\mathbf{y})$ and $P(z|\hat{\mathbf{y}})$:

$$P^*(z|\mathbf{y}) = (1-\alpha)P(z|\mathbf{y}) + \alpha P(z|\hat{\mathbf{y}}) \quad 0 \leq \alpha \leq 1 \tag{5.8}$$

α is the interpolation coefficient. There are also other smoothing methods, e.g., Good-Turing, back-off. Good overviews can be found in the work of Chen and Goodman [19] and Xiao et al. [131].

N-gram models are widely used in statistical natural language processing. In speech recognition, phonemes and sequences of phonemes are sometimes modeled using an n-gram distribution. N-grams can also be used for sequences of words or, in fact, for almost any type of data. They have also been very successful as in genetic sequence search.

The reasons why n-gram models achieve wide usage are, firstly, the parameters can be easily estimated automatically from a training sample, secondly, they can be combined with other statistical recognition systems, thirdly, the models capture the word frequency in the syntactic, semantic, and pragmatic restrictions of the language. N-grams are criticized because they lack an explicit representation of long range dependencies. While it is true that the only explicit dependency range is $(n-1)$ tokens for an n-gram model, in practice, only small values of n are used. Long range dependencies can be achieved by incorporating some kind of local state. This is often done using hand-crafted state variables that represent, for instance, the position in a sentence, the general topic of discourse or a grammatical state variable. Another criticism that has been put forward is that Markov models of language, including n-gram models, do not explicitly capture the syntactic-semantic distinction in languages. N-grams and related language models opt for a fairly pragmatic approach to language modeling that emphasizes empirical results over theoretical purity. There are other models focusing on the modeling of the syntactic rules. In the next subsection, the grammatical models will be introduced.

5.2.2 Grammar Models

In the previous subsection, a pure statistical model for representing the sequence of strings was presented. But it is not concerned with the "rules" underlying the generation of the sequence of the strings – the grammar.

In computer science and linguistics, a formal grammar, or sometimes simply grammar, is a precise description of a formal language — that is, of a set of strings. The two main categories of formal grammar are generative grammars, which are sets of rules for how strings in a language can be generated, and analytic grammars, which are sets of rules for how a string can be analyzed and to determine whether it is a member of the language. Our interests lie on how to write those strings in the set. Therefore, the generative grammar

is our focus. The classic formalization of generative grammars is first proposed by Noam Chomsky in the 1950s [21; 20]. Formally, a grammar G consists of four components:

Symbols: Every sentence consists of a string of characters, which are also called primitive symbols, terminal symbols or letters, taken from an alphabet A.

Variables: These are also called nonterminal symbols, intermediate symbols, or occasionally internal symbols, and are taken from a set I.

Root Symbol: The root symbol or starting symbol is a special internal symbol, the source from which all sentences are derived. The root symbol is taken from S.

Productions: The set of production rules, rewrite rules, or simply rules, denote P, specifies how to transform a set of variables and symbols into other variables and symbols. These rules determine the core structures that can be produced by the grammar. For instance if A is an internal symbol and c a terminal symbol, the rewrite rule $cA \rightarrow cc$ means that any time the segment cA appears in a string, it can be replaced by cc.

Thus, a general grammar can be represented as:

$$G = (A, I, S, P) \tag{5.9}$$

The language generated by grammar G, denoted by $L(G)$, is defined as all those strings over G that can be generated by starting with the start symbol S and then applying the production rules in P until no more nonterminal symbols are present.

Consider the grammar G where $I = \{S, B\}$, $A = \{a, b, c\}$, S is the start symbol, and P consists of the following production rules:

1. $S \rightarrow aBSc$
2. $S \rightarrow abc$
3. $Ba \rightarrow aB$
4. $BB \rightarrow bb$

Some examples of the derivation of strings in $L(G)$ are:

$S \Rightarrow_2 \mathbf{abc}$

$S \Rightarrow_1 \mathbf{aBSc} \Rightarrow_2 aB\mathbf{abc}c \Rightarrow_3 a\mathbf{aB}bcc \Rightarrow_4 aa\mathbf{bb}cc$

$S \Rightarrow_1 \mathbf{aBSc} \Rightarrow_1 aB\mathbf{aBSc}c \Rightarrow_2 aBaB\mathbf{abc}cc \Rightarrow_3 a\boldsymbol{aB}Babccc \Rightarrow_3 aaB\boldsymbol{aB}bccc$
$\Rightarrow_3 aa\boldsymbol{aB}Bbccc \Rightarrow_4 aaaB\boldsymbol{bb}ccc \Rightarrow_4 aaa\boldsymbol{bb}bccc$

In above, $L \Rightarrow_i R$ means "L generates R by means of production i" and the generated part is each time indicated in bold. This grammar defines the language $L = \{a^n b^n c^n | n \geq 1\}$ where a_n denotes a string of n consecutive a's. Thus, the language is the set of strings that consist of 1 or more a's, followed by the same number of b's, followed by the same number of c's.

When Noam Chomsky first formalized generative grammars, he classified them into types now known as the Chomsky hierarchy [20].

Type 0: Free or Unrestricted : Free Grammars have no restrictions on the rewrite rules and thus they provide no constraints on structure of the strings they can produce. Because of their complexity in learning, this type of grammar has found little use in pattern recognition.

Type 1: Context-Sensitive A grammar is called context-sensitive if every rewrite rule is of the form: $\alpha Y \beta \rightarrow \alpha x \beta$. α, β and x are any strings made up of intermediate and terminal symbols, Y is an intermediate symbol. It can be said as "Y can be rewritten as x in the context of α on the left and β on the right".

Type 2: Context-Free A grammar is called context-free if every rewrite rule is of the form: $Y \rightarrow x$. Y is an intermediate symbol, and x is an string of intermediate or terminal symbols. The left-hand side of each production rule consists of only a single nonterminal symbol. Clearly, unlike Type 1 grammar, there is no need for "context" for the rewriting of Y by x.

Type 3: Finite State or Regular A grammar is called regular if every rewrite rule is of the form: $Y \rightarrow y$ OR $Y \rightarrow yZ$. It restricts its rules to a single nonterminal Y on the left-hand side and a right-hand side consisting of a single terminal y, possibly followed by a single nonterminal yZ. Such grammars are called finite state because they can be accepted by a finite-state automaton.

In the above, we know how to form a derivation from a root node to a final sentence. For recognition, the inverse process is employed: that is, given a particular X, find a derivation in G that leads to X. This process is called *p*arsing. The Type 2: context-free grammars and Type 3: regular grammars are most often used because parsers for them can be efficiently implemented. For concrete parsing algorithms, please refer to [43].

When a system is supposed to recognize activities with some predefined context or inherent semantics, pure probabilistic methods are limited with regard to representing the underlying structure. Because of the modeling of the production rules between the symbols, grammar models are good at recognizing rule-based activities. For the probabilistic finite state machine representations, it is difficult to add new states if the training has been done. Grammar allows us to use a single, compact representation for longer dependencies between the symbols, which is the point for criticizing HMMs. However, a grammar also has disadvantages. It is very domain dependent. It is very difficult to learn by machines without human supervision. It is also sensitive to the errors in the strings. Normally, a parser only gives the result whether a string can be successfully parsed by a grammar. But spurious symbols in the input can generate ungrammatical strings, which cause the parsing algorithm to fail.

In order to avoid the $\{1, 0\}$ parser decision, which is sensitive to the errors in the symbol sequence, an important extension of context-free grammar has been made: the stochastic context-free grammar (SCFG), which is also named probabilistic context-free grammar, PCFG). It is a context-free grammar in which each production is augmented with a probability just like hidden Markov models extend finite state machines. By attaching a probability to a particular sequence of events, it provides a quantitative basis for ranking

parses. With this augmentation, SCFGs have found widespread applications in the recognition of rule-based activities. Bobick and Ivanov used it for the tasks, like recognizing a simple gesture – drawing a square in the air, which needs significant effort to solve using only the statistical pattern recognition techniques [9]. It has also been used for segmenting the musical conducting gestures based on different beat patterns [9], recognizing the multitasked activities in poker games [79], and classify the tea ceremonies [132]. Because the grammar models are relatively sensitive to the errors in the symbol sequences, the intuitive solution is to minimize the errors appearing in the symbol generation process, which is always the low level processing for basic event detection in vision-based systems. For example, Yamamoto et al. [132] segmented the actions based on acceleration, which could produce actions corresponding to a symbol with little error. But the errors in symbol sequences are inevitable. Bobick and Ivanov handle the insertion and substitution error by modifying the grammar that the parser accepts input that could generate a fatal syntax error [9]. Rather than allowing errors by extending grammar, Moore and Essa tried to recover the errors by multiple hypotheses which are derived from the type of errors [79].

5.3 Layered Representation of Manipulative Task

As mentioned in Section 2.2.2, the manipulative task recognizer is designed for a mobile household robot. As far as we know, there are no mobile robots which have such a system integrated. But the recognition of complex human activities is not new. This topic has been investigated for a long time in areas like visual surveillance, and ubiquitous computing. By recognizing the tasks the people want to fulfill or the abnormal behaviors, appropriate suggestions can be given by the intelligent systems. For example, a vision system is developed by Gao et al. [39] to measure feeding difficulties in nursing home residents with severe dementia. In the work of Fukuda et al. [37], a robot companion worked as a cook assistant by sensing the manipulations on different objects. A common point in these systems is that the layered structure is chosen to model the activities. The reason is two fold. On one side, the complex activities themselves are intuitively understood by humans as final goals which are achieved by completing several subgoals. On the other side, the variance and dependency in the longer observation of complex activities are hard to model by a single, flat mathematical model. The main differences among these models are where the symbol grounding happens and how the symbols are connected together. In this section, the structure and the mathematical representation of our manipulative task model will be presented.

5.3.1 Two-Layer Structure

Understanding the modular organization of biological movement are outstanding challenges both in biology and robotics. In the 90s, several experiments done by biologists have proved the existing of motion primitives [94; 104], which are elementary movements constituting a motor vocabulary composable into a broad movement repertoire. The theory of motor primitive has inspired the behavior-based motion control of robots. Following this idea, the concept of primitive is also widely used in vision-based human activity recognition. However in most cases, the "primitives", which the vision systems recognize, are no longer the motor primitives but a set of basic elements which construct the target activities. For example, to recognize the activities in an indoor environment, the movement

of human between two landmarks in the environment are chosen as primitives by Nguyen et al. [86] and Yin, Chai and Yang [134]. Hongeng and Nevatia used the state of relative movement like "approach", "stop", and "leave" as primitives to recognize more complex interactions between several agencies like "theft at phone booth" [49].

Because people extensively use their hands to interact with the environment, the manipulative gestures are also targeted by many vision-based recognition systems. In the work of Jo, Kuno, and Shirai [55], the manipulative hand gesture is modeled as a sequence of transitions between manipulative states, such as "grasp", "move", and "release". The transitions are triggered when predefined conditions are satisfied. Because the conditions are defined according to known situational knowledge, this recognition system worked well in a restricted human-computer interaction scenario [55]. In order to teach everyday manipulative tasks to a robot under the "learning from observation" framework, Ogawara et al. [87] proposed a technique to estimate essential interactions to complete a manipulative task from several demonstrations of it. The essential interactions, also named primitive actions in their paper, are made up of three properties: the grasped object, the target object and the set of relative trajectories. However, manipulations need not to have two objects involved. For example, "type on the keyboard" is a kind of manipulation. It involves no object in hand. Moreover, the objects in hand are hard to detect in vision-based systems because of the occlusion.

In our modeling, the interpretation of manipulative gestures has two layers. The structure of the model is shown in Figure 5.1. At the bottom, the low level image processing extracts the hand trajectory and objects from image sequences. For each object detected, a processing thread is created. The hand trajectory is represented differently as relative movement to the object in different threads. Then, the object-oriented manipulative primitives can be detected in the threads (see also Section 3.3.1 and 3.3.2). Besides the argument that the manipulative primitive are object-specific, the other reason to use the object-oriented primitive is that this definition also benefits the lower level processing and higher level interpretation. In lower level processing, the object-centered trajectory representation supposes a static object in focus. It avoids the detection of an object in hand and the tracking of moving objects. Without the parallel processing assumed for object-specific primitive, the trajectory interpretation has much ambiguity. For example, a movement from a cup to a sugar box can be "leaving cup" or "approach sugar". If these two objects are near to each other, it is hard to find a point to separate this trajectory into two meaningful segments. Assuming parallel threads for each single object, the hand trajectory has no such problem. The manipulative primitives detected in different threads are sent upward to the task level. In the task level, the manipulative primitives detected from all threads are put together and processed in temporal order. In next subsection, the mathematical model which matches a manipulative task to a primitive sequence will be put forward.

5.3.2 Task Model

As described in the previous section, the task layer takes the manipulative primitives detected from all threads. When a sequence of manipulative primitives is available, the next problem is which manipulative task was performed. A sequence of manipulative

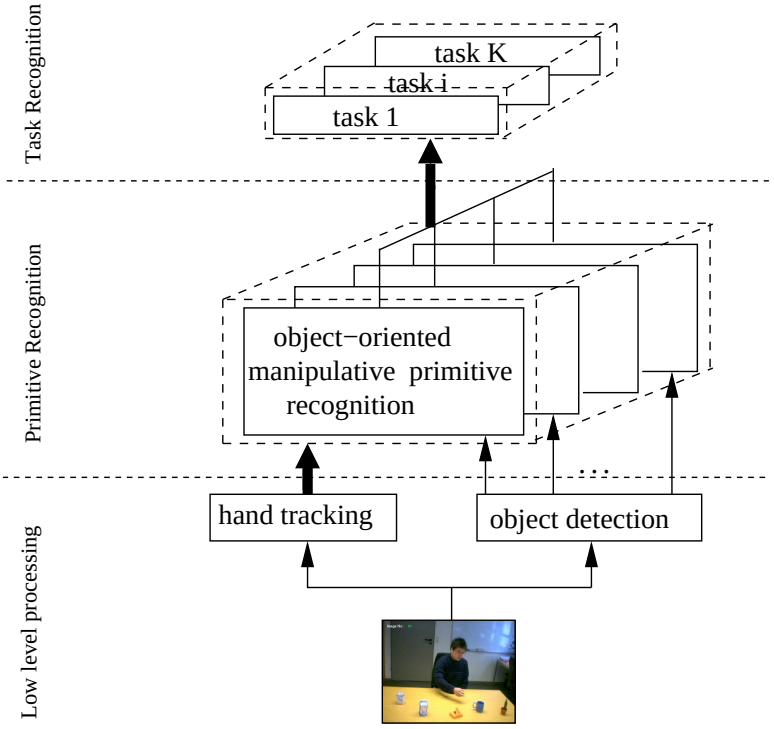


Figure 5.1: The layered system architecture

primitives can be thought of as a sequence of symbols. The models for coding symbol sequences have been described in Section 5.2. Before choosing an appropriate model, the properties of the task recognition layer must be analyzed. First of all, the inputs are the output from primitive layer. The errors of the detection are inevitable. The task model has to take the recovering of the errors into consideration. Secondly, the performance of the manipulative task is relatively free. This means that the ordering of the primitives in the sequences is not fixed. For example, when a person wants to prepare a cup of coffee, she/he could either add sugar firstly or add some milk at first. Third, a task can be performed differently or performed with certain personality. For a personal robot, it should have the capability to adapt its behavior or learn different manipulative tasks by its own observation.

Therefore, the manipulative tasks are modeled as a first-order Markovian process (bigram model) by us, which is the same as Moore's definition [80]. Compared with a grammar model, n-gram models are easy to learn. It is less sensitive to the errors in the symbol sequence. Although this assumption violates certain long-term domain dependencies, it is an efficient and practical way to deal with task knowledge.

The task models share a set of possible manipulative primitives. The model Λ_i for a manipulative task i contains the transition matrix A_i, the initial probability Π_i, the terminal probability E_i, and a threshold th_i. Suppose the result from the manipulative primitive recognition is the sequence P_o. The acceptance of a task $\Lambda_i = (\Pi_i, A_i, E_i, \mathrm{th}_i)$, is defined relative to a random model $\Lambda_r = (\Pi_r, A_r, E_r)$, which is similarly defined as a task

model but includes no associated threshold and is learned from all the training data from different tasks. The similarity of the sequence and a task model $s(i, P_o)$ is calculated as:

$$\begin{aligned} s(i, P_o) &= \log(\frac{p\{P_o|\Lambda_i\}}{p\{P_o|\Lambda_r\}}) \\ &= \log(\frac{p\{P_o|\Pi_i, A_i, E_i\}}{p\{P_o|\Pi_r, A_r, E_r\}}) \end{aligned} \tag{5.10}$$

The task decision $d(P_o)$for recognition is

$$d(P_o) = \begin{cases} \arg\max_i \{s(i, P_o)|s(i, P_o) > \text{th}_i\} \\ \text{Null} \end{cases} \tag{5.11}$$

In everyday life, there are also many manipulations which the robot doesn't understand or just have no clear purpose. In order to avoid classifying these manipulations into known tasks, Eq. 5.11 also takes the possible rejection into consideration. Only if the similarity measurement between a manipulative sequence and a task model is higher than the associated threshold of that task model, this task model can be selected as a candidate.

5.4 Coupling of Top-Down and Bottom-Up Processes

Figure 5.1 shows the layered structure of the manipulative task model. In it, the processing goes from bottom to top, for each detected object a thread is created that consists of a trajectory segmentation, a feature computation, and an HMM-based recognition step. All three steps are performed differently for each object in parallel and the hand trajectory information is passed to each object-centered processing thread. Because of the object-specific primitive definition and its parallel processing for each affected object, the system confronts an attention problem when there are many objects appearing in the scene simultaneously.

This problem also appears in many other recognition tasks. In the review paper of Tsotsos [118], he gave a definition of attention in visual perception:

Definition 5.1 *Attention is a set of strategies that attempts to reduce the computational cost of the search processes inherent in visual perception.*

The attention mechanism can be divided into two categories: stimulus-driven and goal-directed. In the work of Michalke et al. [77], both of the mechanisms are used for a driver assistance system. In the stimulus-driven channel, a saliency map of the scene is calculated based on different low-level visual features including orientation, intensity, and motion. In the goal-directed channel, a saliency map is also generated according to the current context. For example, while driving at high speed, the central field of the visual scene becomes more important than the surrounding.

In the manipulative task recognition, a top-down process is introduced in the system architecture (see Figure 5.2). It is a goal-directed attention mechanism. The difference is it

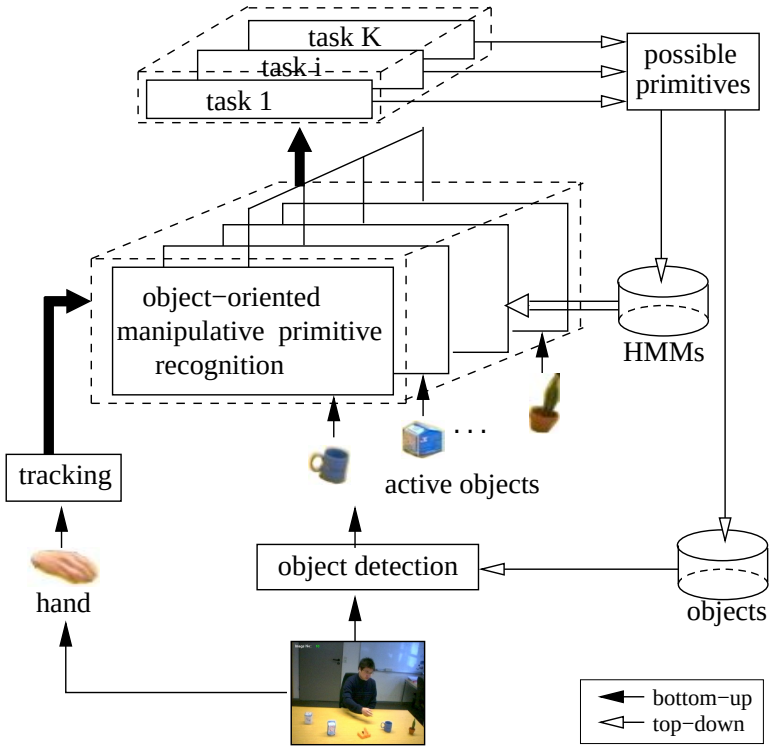


Figure 5.2: The system processing flow combining both top-down and bottom-up processes

is initialized by the current state of robot but it is continued based on the recognition history. It means that the possible primitives coming next are predicted on the ground of the active task models and the previous results from the manipulative primitive recognition. The mechanism combines the bottom-up and top-down processes in a loop.

This prediction is similar to the computation of a lookahead symbol in parsing strategies. The possible word pair transitions of a task model are extracted from the primitive transition matrix by a frequency threshold. For the prediction step different parsing alternatives are considered during the HMM matching process. For all primitives that achieve an end probability $P_{\text{end},t}(p_i) > 0$ a lookahead symbol is generated. If a primitive has been recognized this primitive is eliminated as a lookahead symbol. Because the predicted action primitives are specific for certain object types, the set of the next possibly manipulated object types can be calculated and be passed to the object detection component. This realizes a task driven attentional cue for early processing steps of the system. Additionally, the expectations from the predicted action primitives are used to restrict the HMMs applied in the PF based matching process.

5.5 Task Recognition in an Office Scenario

The evaluation assesses the overall system performance. In our experiment, a scenario in an office environment is set up as shown in the images in Figure 5.4. A person is sitting behind a table and manipulates the objects that are located on it. She or he is assumed to perform one of three different manipulation tasks:

- *water plant*: consists of take cup, water plant, put cup
- *prepare tea*: consists of take/put cup, take tea can, pour tea into cup, put tea can;
- *prepare coffee*: consists of take/put cup, take milk/sugar, pour milk/take sugar into cup, put milk.

A manipulative task consists of the manipulative primitive sequence. However, the ordering of the sequence is neither pre-determined nor completely fixed. For example, some people may take sugar before taking milk, some will do it the other way around. But there probably will be an ordering between taking the cup and the watering action which needs to be learned from the data.

Table 5.1: The index of the manipulative primitive in the experiment.

Objects	tea		milk		sugar	cup			plant
Primitives	take	put	take	put	take	take	put	pour	water
Index	1	2	3	4	5	6	7	8	9

For learning the possible transition pairs of each task model, the data set is divided into the set of 20 observation sequences, and a set of 16 sequences that are used for a one-leave-out experiment. Thus, each task model is learned from 35 task sequences in each experiment. Table 5.1 shows the coding of manipulative primitives in a transition matrix in our experiment. The matrix on the left side of Figure 5.3 is a primitive transition matrix of "prepare tea" task which is learned from training data. The cooresponding word pair transitions for task level prediction shown on the right side of Figure 5.3 are extracted by the threshold 0.2, which is set empirically.

Figure 5.4 shows the evolution of the active manipulative primitives throughout the recognition of this "prepare tea" task. Here, The X-axis represents the numbering of image frames. On the Y-axis, each primitive has a corresponding horizontal line. The shadowed bars on the horizontal lines mark the active period of each primitive in this recognition process. It can be found that the two primitives "put cup" and "water plant" are ready

$$\begin{bmatrix} 0.006 & 0.024 & 0.006 & 0.006 & 0.006 & 0.114 & 0.060 & 0.772 & 0.006 \\ 0.083 & 0.083 & 0.083 & 0.083 & 0.083 & 0.083 & 0.250 & 0.167 & 0.083 \\ 0.100 & 0.200 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.1000 \\ 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.200 & 0.100 & 0.100 & 0.100 \\ 0.111 & 0.111 & 0.111 & 0.111 & 0.111 & 0.111 & 0.111 & 0.111 & 0.111 \\ 0.155 & 0.005 & 0.005 & 0.005 & 0.005 & 0.027 & 0.749 & 0.043 & 0.005 \\ 0.730 & 0.184 & 0.006 & 0.006 & 0.006 & 0.035 & 0.012 & 0.017 & 0.006 \\ 0.007 & 0.912 & 0.017 & 0.007 & 0.007 & 0.007 & 0.027 & 0.017 & 0.007 \\ 0.111 & 0.111 & 0.111 & 0.111 & 0.111 & 0.111 & 0.111 & 0.111 & 0.111 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 5.3: From transition to pairwise grammar

to be recognized from the beginning though none of them will be performed as the first primitive of these three tasks. This is because that the task models are learned with the primitive sequences which may contain the deletion errors of the primitive "take cup". It is also means that a task could be correctly recognized even if the first primitive "take cup" has been missed by the primitive detection. Furthermore, Figure 5.5 shows the end

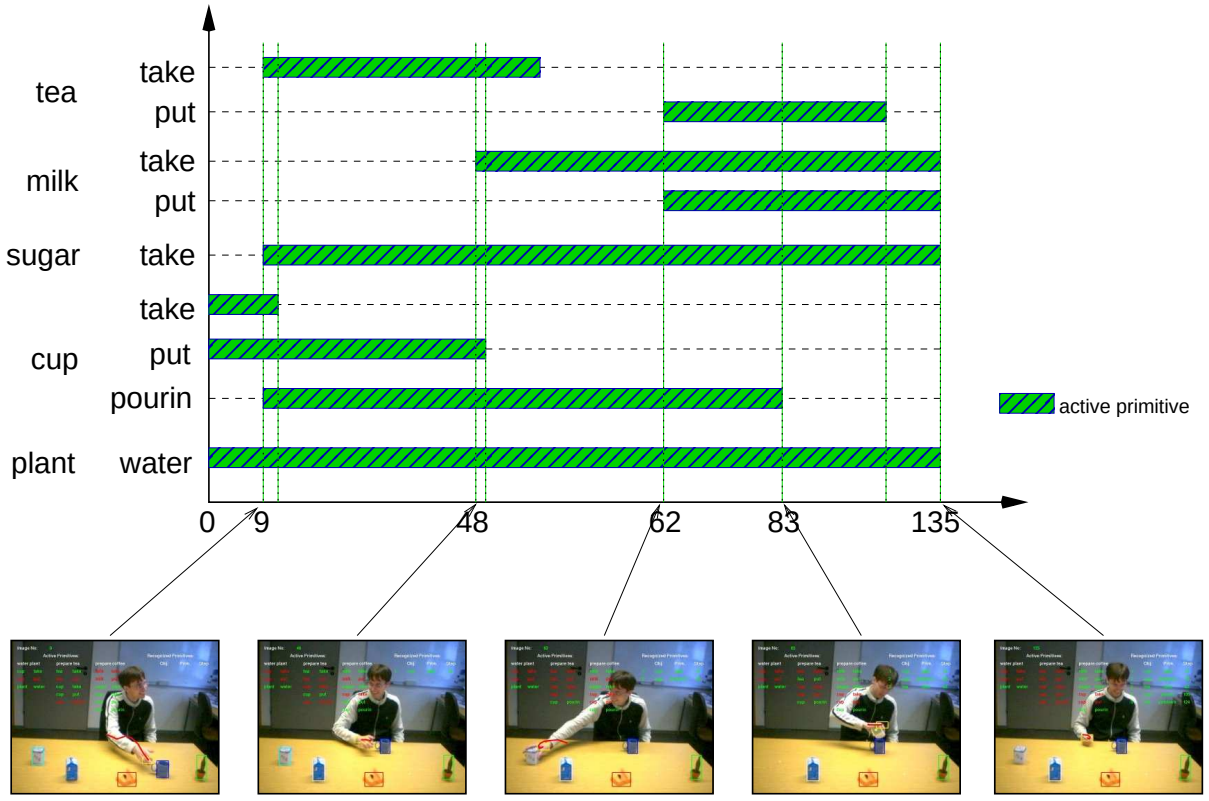


Figure 5.4: The active primitives in a "prepare tea" task

probabilities of different manipulative primitives in a prepare tea task. The X-axis has the same meaning as in Figure 5.4. The horizontal line above zero is the recognition threshold and the temporal periods which are colored indicate that the hand is in the object vicinity at that moment.

The task recognition results are shown in Table 5.2. It compares the results between the processes with (TD) and without (no TD) the top-down attention processing. The error rates clearly show significant drops between 16% and 28%. The error rates of tasks "prepare tea" and "prepare coffee" are relatively high. But most of them are caused by the rejection of these tasks rather than substitution errors (Sub.). The reason why the task "prepare tea" has relatively high substitute error has several aspects. Firstly, the primitives from other tasks are allowed in the task models because of the errors in the primitive detection, even they have only small probabilities. Secondly, the primitive sequence in the "prepare tea" task is fixed, relatively short and shares several primitives with "prepare coffee" task. If some primitives from "prepare coffee" task are detected in this task, these insertion errors could cause the task "prepare tea" to be recognized as

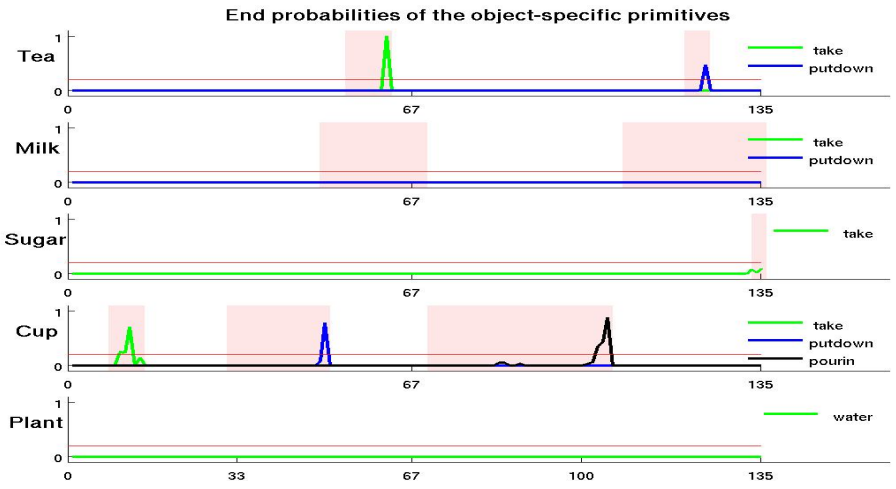


Figure 5.5: The end probabilities of the manipulative primitives in a "prepare tea" task

"prepare coffee". The top-down processing does not only improve the recognition rate but also decreased the computational load. The processing time for a 180-frame "prepare coffee" sequence with top-down is 54s running on MATLAB, which is much lower than the 86s needed by the pure bottom-up processing.

Table 5.2: The recognition results of the manipulative tasks with and without top-down processing.

Name	Num.	FN(%, TD)	FN(%, no TD)	Sub.(%, TD)
water plant	16	5.8 ± 2.1	8.1 ± 3.4	1.0 ± 1.2
prepare tea	16	17.6 ± 6.4	22.3 ± 2.2	6.5 ± 3.8
prepare coffee	16	11.6 ± 5.2	13.9 ± 3.1	0.7 ± 1.0

5.6 Hierarchical Representation

Although a two-layer structure is used in the manipulative task model, it has the potential to be extended to a hierarchical structure. The hierarchical modeling of human activities has appeared in the work of other researchers. A hierarchical hidden Markov model (HHMM) originally proposed by Fine et al. [28] is used to solve the problem of representing dynamic, multi-level processes. Aiming at modeling the hierarchical structure of human intentions and activities, Bui [13] introduced the abstract hidden Markov model (AHMM) to represent hierarchical indoor activities. For example, "go to printing" includes a sequence of actions such as "go to computer", "go to printer" and "go to paper store" if the printer is out of paper. An online task recognition is realized by a Rao-Blackwellised particle filter. Liao [72] gives an example of learning the structure and parameters of an AHMM model by expectation maximization (EM). His system infers a

person's transportation mode and target in city traffic from GPS signals. The AHMM can be thought of as a kind of HHMM which is an extension of the hidden Markov model to include a hierarchy of the hidden states. In order to achieve the efficient sharing of the substructures, a lattice structure is put forward in [14].

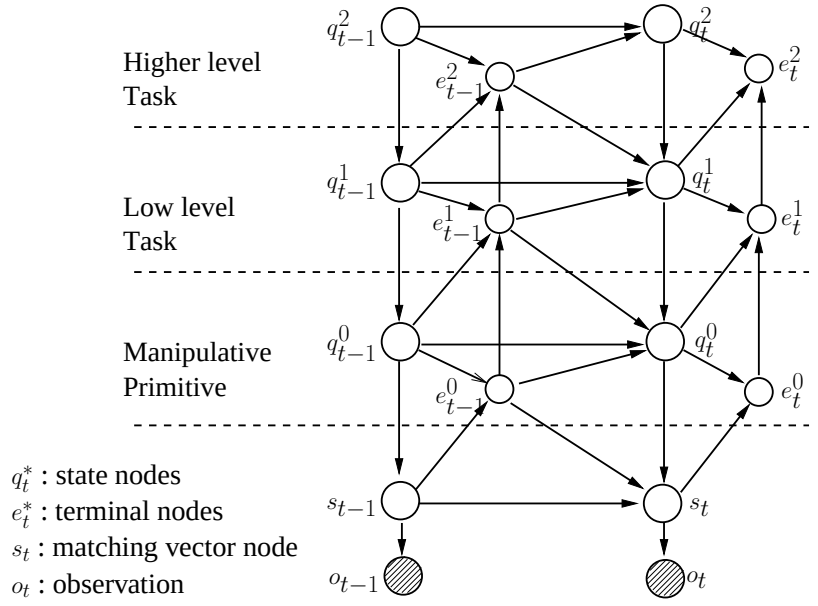


Figure 5.6: The DBN representation of the hierarchy structure for recognizing manipulative activities

Figure 5.6 shows the DBN representation of the hierarchical model for manipulative task recognition. The model has three layers: manipulative primitive, low level task and higher level task. The names of the layers – low level and higher level task – give only the idea of hierarchy. They need to have concrete definition in applications. The variables q_t^0, q_t^1, and q_t^2 are used to represent the state of these layers at time t. They are called state nodes. For each level $d \in \{0, 1, 2\}$, the collection of possible states of level d is denoted as Q^d. For each state $q^d \in Q^d$, $d > 0$, the children of q^d are represented as $\mathrm{ch}(q^d)$ with $\mathrm{ch}(q^d) \in Q^{d-1}$. In the same way, the parents of q^d, $d < 2$ are represented as $\mathrm{pa}(q^d)$ with $\mathrm{pa}(q^d) \in Q^{d+1}$. On each level d, there is a terminal node e_t^d linking to the state node q_t^d. It is a binary node representing whether the current state ends at time t. If $e_t^d = 0$, that means the state q_t^d will continue. Otherwise when $e_t^d = 1$, the state q_t^d ends here and a new state will be initialized in the next time step. A terminal node equals one only when the value of the terminal node of the lower level is one because a state will not end when its sub-state is still in execution. So the states of the end nodes are inferred with a bottom-up sequence.

There are three kinds of probabilities for parameterizing the discrete part of the model. Suppose p is a state of level d ($d \in \{0, 1, 2\}$), and the states i, j are two lower level states $i, j \in \mathrm{ch}(p)$, we define:

- $\pi_i^{d,p}$ as the initial probability of the child i given the parent is p,

- $\mathbf{A}_{i,j}^{d,p}$ as the transition probability from child i to j given the parent p,
- $\mathbf{A}_{i,end}^{d,p}$ as the terminal probability that state p terminates given the current child is i.

The conditional probabilities $P(v|\text{parent}(v))$ in the model are defined from the parameters of the HHMM and effected by the different states of the terminal nodes. Here, parent(v) denotes the parent nodes of v in the directed Bayesian network. The probability for the termination of state p is inferred as follows:

$$P\{e_t^d = 1|q_t^{d-1} = i, q_t^d = p, e_t^{d-1}\} = \begin{cases} \mathbf{A}_{i,end}^{d,p} & \text{if } e_t^{d-1} = 1 \\ 0 & \text{if } e_t^{d-1} = 0. \end{cases} \tag{5.12}$$

The transition probability between the states of two consecutive time steps in the same level is calculated as follows:

$$P\{q_t^d = j|q_{t-1}^d = i, q_t^{d+1} = p, e_{t-1}^{d:d+1}\} = \begin{cases} \delta(i,j) & \text{if } e_{t-1}^{d:d+1} = 00 \text{ (both continue)} \\ \mathbf{A}_{i,end}^{d-1,p} & \text{if } e_{t-1}^{d:d+1} = 10 \text{ (child continues)} \\ \pi_j^{d-1,p} & \text{if } e_{t-1}^{d:d+1} = 11 \text{ (both end)} \end{cases} \tag{5.13}$$

With Eq. 5.12 and Eq. 5.13, the probability of a node state in the discrete part of the HHMM can be calculated based on the states of its parent nodes. The probability $Pr\{q_t^2|o_{1:t}\}$ of the top node state given the observation sequence represents the likelihood of the current manipulative task in execution. The online state estimation of this probability can be realized by a particle filter [83; 14].

This hierarchical model has been applied to recognize office actions [68] using a simpler primitive modeling. Before a performance evaluation of this model based on the current primitive definition can be performed, a correct hierarchy of the manipulative tasks must be built up firstly. However, as far as we know, there is no literature on the theoretical analysis of the structuring of human manipulative tasks providing an easy solution.

5.7 Summary

In this chapter the two-layer system architecture used for the recognition of manipulative tasks is introduced. In this two-layer task model, the task layer receives a sequence of manipulative primitives as input. For such a symbol sequence modeling and recognition problem, the n-gram model is chosen by us other than grammar model because it provides an easy way to recover the primitive detection errors and learn the model from training data. However the decision of recognition is not only based on the probability of the primitive sequence given the N-gram task model. The tasks which have a longer sequence of primitives have smaller probabilities because the transition probabilities in N-gram models are smaller than 1. Therefore a random model is introduced by us. The probability of a primitive sequence given the random model is used to scale the probability calculated based on the task model.

Because of the object-oriented approach, our system is confronted with the attention problem when there are lots of object candidates in the image. We combined a top-down process with the bottom-up process. This processing loop utilizes the task-level prediction of possible primitives in order to restrict the object types possibly detected as well as the action primitives possibly recognized. It forms an attention system of the robot. From the results of a task recognition experiment in an office scenario, it is shown that the compound top-down, bottom-up processing not only improved the recognition rate but also decreased the computational load. In the last section, a hierarchical model is presented to show the extensibility of the system structure.

6. Manipulative Task Learning

For a social robot, the ability of learning tasks via human demonstration is very crucial. Nowadays, a lot of research endeavor is focusing on the development of a robot whose task is to serve humans as a companion in their daily life. In this case, the robot is not only considered as a ready-made device but as an artificial creature, which improves its capabilities in a continuous process of acquiring new knowledge and learning skills. However, robot learning also has different focuses in different fields of research. In robotics, learning has been always considered in frameworks of Learning by Imitation or Programming by Demonstration (PbD) [111; 26], which is a technique for teaching a computer or a robot new behaviors by demonstrating the motions instead of programming it through machine commands. The demonstrated motions are mapped to a sensory based representation according to the embodiment of the robot which enable it to reperform these motions. In the AI community, much of the work has concentrated on the high-level planning and conceptual representation of skills and state changes using propositional or first-order logic. When the task learning is concerned, the two domains in robot learning can be put in a framework and build up a learning hierarchy. A set of motion primitives are obtained by the robot through the learning of motion control. The conceptual representation of higher level tasks codes the correct sequence for the primitive execution in a task.

For the work presented here, we consider a scenario in which we want to enable a robot to learn and then understand complex domestic manipulative tasks. Therefore, firstly, it is not our purpose in this chapter that the robot learns motion primitives from demonstrations. Secondly, our focus also does not lie on the learning for motion planning but learning for recognition. Both of them are kinds of higher level modeling. The difference is the former tries to extract essential primitives from the task demonstrations but the latter finds the variance in the demonstrations.

Following the idea of stack generalization (see Section 5.1), the task learning can be separated to different layers, e.g., primitive layer and symbol sequence layer. Our work in this chapter is a high-level learning with background knowledge because it is supposed that the robot has some basic recognition capabilities of manipulative primitives. That means the models of object-specific manipulative primitives are trained already and they

are relatively stable throughout different tasks. The primitive detector in the lower layer turns the observation of a manipulative task into a sequences of primitives and feeds them into the learning process as input. What's worth mentioning is that the primitive detection is not perfect. Consequently, both data for task learning and recognition could have possible deletion, insertion, and substitution errors in them.

For a household robot, it is natural and convenient for users that they only have to demonstrate the robot the manipulation once or very few times intentionally, and the robot could recognize it afterwards or even update the task knowledge by its observation. Most current approaches suffer from either the demanding of the huge amount of labeled training data, or the limited recognition cabability caused by very domain-specific modeling. In this chapter, a semi-supervised incremental task learning method will be presented [70].

To present our approach, several task models and learning frameworks will be discussed in the next section. Then, it will be introduced how the robot starts the learning by a few labeled training samples and optimizes the models afterwards without the need of a human pre-labeling. With respect to unseen tasks, a solution is presented in Section 6.3 that the system collects and filters the rejected tasks in a buffer which is used to learn new models in an unsupervised way. After that, the results of evaluation will be shown in Section 6.4.

6.1 Task Learning for Human-Robot Interaction

Robots will inevitably become a part of our daily lives. Serving as helpful assistants to untrained users in the home or office, robots should be able to understand what the user is doing and help the user by performing some tasks. However, that a task can be recognized by robots doesn't mean that it can be performed by robots. In our opinion, the task learning of robots can be sorted into two categories: learning for recognition and learning for motion control. Although the hierarchical structure of the task model is used in both types of learning, the concrete modeling and learning strategy on the layers in the hierarchy have much difference between these two types.

Aiming to teach a robot how to perform everyday manipulation tasks, Ogawara proposed a task model which is a sequence of essential interactions [87]. In his definition, an interaction consists of two objects: the grasped object, the target object and the relative motion between them. The relative motions are clustered and represented by an HMM [88]. They argue that an interaction sequence of an observation can have both essential and non-essential interactions. By aligning the interactions of multiple observations of a task using dynamic programming, they could extract the essential interactions for this task. More focusing on the causality among the subtasks, Zoellner firstly has the idea to represent a task by precedence graph [138], which is later on further developed by Pardowitz and named task precedence graph (TPG) [92]. A task called macro-operators (MO) is a directed graph with a set of subtasks as basic elements, which are elementary operators (EO). The directed connection between two subtasks indicates a task execution rule which must be complied with.

It can be found from the work above that the essential primitives and their sequences are important for the correct re-performing of the tasks done by a human. The robust

extraction of these primitives are assured by having particular sensoring equipment, such as dataglove or a stereo vision system with multiple cameras. However, we are in a recognition scenario, where the tasks are only to be re-recognized but not repeated. Therefore, we are trying to seek a good decision boundary between two tasks other than finding their core primitive. In this scenario, the observational conditions are normally worse, e.g., using a low-cost mono-camera for sensing. The errors of primitive detection appear more often than in the learning for motion control. To achieve a good classifier, the variance of the task models must be learned from training data.

The learning in pattern recognition has several general forms: supervised, unsupervised, and reinforcement learning. In supervised learning, a category label is given for every entry of the training set. This form is straightforward and popularly used in the learning of human actions. For example, Liao used Markov-chain Monte-Carlo (MCMC) algorithm to learn the routines of a person's everyday life, like go to shopping and go to work from GPS data [72]. Expectation-Maximization (EM) is an iterative parameter optimization algorithm which is often used in supervised learning [23]. In unsupervised learning, there is no explicit labeling. The system forms "natural" clusters of the input patterns. In the smart homes project, the inhabitant actions of the user are predicted according to task models, which are Markov models that have been generated from an unsupervised clustering [103]. In reinforcement learning, no desired category signal is given; instead, the only teaching feedback is that the tentative category is correct or incorrect. This learning mechanism is mainly used in the learning for motion control in robotics [73].

In practice, the learning needs always a lot of training data. In Rao's work, the training data is equal to the recording which needs 1250 days [103]. However, in some cases, the labeled instances are difficult, expensive, or time consuming to obtain. Meanwhile unlabeled data may be relatively easy to collect, but there has been few ways to use them. Therefore, a learning method named semi-supervised learning has also received much attention recently. It tries to use unlabeled data, together with the labeled data, to build better classifiers [137]. It has found good applications in fields like text classification, object recognition, image retrieval, and interactive image segmentation

Semi-supervised learning is also used in the action learning and recognition. Grossman et al. presented a semi-supervised method for clustering robot experiences [42]. In their work, the robot experiences were represented as a multivariate time series containing several measurements from a set of sensors. The similarities between them were calculated using DTW. The labelled sequences were used to assess the clustering quality. According to the clustering quality, the decision to stop clustering will be made. In order to help the older adults which have cognitive disabilities (such as dementia) and can not complete their activities of daily living, Hoey et al. use a partially observable Markov decision process to monitor a user and to assist the user during each activity [45]. The state of every time slice in their model is factored into three sets of variables, task, attitude, and behavior. However, the labeling of behavior is a labor-intensive task requiring expert knowledge. Therefore, the learning method clusters video sequences in training data in which only the task states are labeled. Thereby, they learn a set of behaviors, and the relationship of the learned behaviors to the task states.

6.2 Semi-Supervised Incremental Task Learning

For a household robot, it may be feasible to group similar activities after observing a certain amount of tasks for many times. But more naturally, users will expect that the robot is able to learn some tasks just from a few demonstrations. However, the variance of errors in the primitive recognition is hard to be discovered from several demonstrations. Chan et al. proposed an activity model which can be learned from one demonstration [18]. In their work, a simple feature vector is used for modeling the interaction primitives, e.g. "approach". The state transitions inside the semantic primitives are modeled by HMMs. The semantics between the primitives can only be learned with prior structural knowledge given by a human.

Semi-supervised learning provides a solution to use limited labelled data together with a large amount of unlabeled data in a learning framework. It has mainly two forms: self-training and co-training. Self-training is a commonly used technique for semi-supervised learning. In self-training a classifier is first trained with the small amount of labeled data. The classifier is then used to classify the unlabeled data. Typically the most confident unlabeled points, together with their predicted labels, are added to the training set. The classifier is re-trained and the procedure is repeated. Note that the classifier uses its own predictions to teach itself. Co-training assumes that features can be split into two sets; each sub-feature set is sufficient to train a good classifier; and the two sets are conditionally independent given the class. Initially, two separate classifiers are trained with the labeled data, on the two sub-feature sets, respectively. Each classifier then classifies the unlabeled data, and "teaches" the other classifier with the few unlabeled examples (and the predicted labels) they feel most confident. Each classifier is retrained with the additional training examples given by the other classifier, and the process repeats.

Because the inputs of the classifiers are manipulative primitive sequences, a semi-supervised training framework with self-training form is chosen by us. The task model is presented in Section 5.3.2. Principally it is a N-gram model. The difference is that not only the parameters for different task models but also for the random model must be learned for the sake of achieving a normalized similarity measurement between models and primitive sequences with various lengths. The semi-supervised learning process is shown as pseudocode in Figure 6.1. It starts from a small set of labeled sequences. Suppose there are n different labels, n manipulative task models will be constructed. The model for task i that is learned from u labeled data is represented as $\Lambda_{i,u}$. $\tilde{A}_{i,u}$ is the matrix recording the counts of all transitions between the primitives which occurred in the labeled data. As we discussed before, non-occurring n-grams could cause zero probabilities during the matching. Several smoothing methods for n-gram model have been briefly presented in Section 5.2.1. Here, we use a simple "adding one" method. Consequently, the matrix $A_{i,u}$ is the resulting matrix $1 + \tilde{A}_{i,u}$ normalized in column. $\Pi_{i,u}$, $E_{i,u}$ and the parameters in Λ_r are computed in the same way. But the random model Λ_r is learned using all the sequences from different tasks. Given Λ_r and Λ_i, the similarity measurements of the labeled sequences from one task can be calculated according to Eq. 5.10. Then, $\mathrm{th}_{i,u}$ is set as the minimum of them and the corresponding sequence is saved in memory as the instance with minimal matching P_i^{th}.

/∗ ***Supervised Learning*** ∗/
1. construct $\Lambda_{1\ldots n}$ and Λ_r from labeled data
2. find P_i^{th} for each task model Λ_i $(i = 1 \ldots n)$

/∗ ***Unsupervised Learning*** ∗/
for each new unlabeled sequence P_m **do**
  1. update random model Λ_r
  2. compute $d(P_m)$ (see Eq. 5.11)
  if $d(P_m) = j$ **then**
    update task model j
  else $\{d(P_m) = \text{null}\}$
    $i^* = \arg\max_i(s(i, P_m))$
    if $s(i^*, P_m) \geq Th_o$ **then**
      1. update task model Λ_{i^*}
      2. choose new $P_{i^*}^{\text{th}}$
    else $\{s(j^*, P_m) < Th_o\}$
      /∗ *reject* P_m *as unseen task* ∗/
      insert P_m into buffer

Figure 6.1: Pseudocode of semi-supervised task learning

When an unlabeled sequence P_m is perceived, the random model is updated first. As a consequence, the $\text{th}_{i,u}$ is renewed to $\hat{\text{th}}_{i,u}$. Afterwards, the decision of P_m is made based on Eq. 5.11. There are two possibilities.

- Case 1: The sequence P_m is sorted into task j. The task model Λ_j will be updated.
- Case 2: The decision is Null which means no $s(i, P_m)$ greater than $\hat{\text{th}}_{i,u}$.

Because this decision is only made according to the current task models, in order to group similar sequences and update a task model during the learning process, a general lowest matching threshold Th_o is introduced, which defines the allowed loosest match between the sequence and its task model. So when the process entered case 2, the best matching task model $i^* = \arg\max_i(s(i))$ is computed. Dependent on the comparison of $s(i^*, P_m)$ and Th_o, the case 2 is divided into two subcases.

- Case 2.1: If $s(i^*, P_m)$ is greater than Th_o, the sequence is labeled as i^*. The task model Λ_{i^*} will be updated and the worst matching sequence of this model will be evaluated again.
- Case 2.2: All $s(i, P_m)$ are less than Th_o. This means it is far away from known models and just rejected as an unseen task.

A new sequence is labeled when the process goes into case 1 or case 2.1. The corresponding task model will be updated. Suppose $\tilde{A}_{i,\text{one}}$ represents the transition of the primitives in

the new sequence, $A_{i,u+1}$ is the normalized $1 + \tilde{A}_{i,u} + \tilde{A}_{i,\text{one}}$. The update of Π_i, E_i is in the same vein.

When the process has entered the case 2.1, the task threshold will also be updated. The similarity measurements of the sequence in memory and the new observation given the updated model are calculated again, the task threshold $\text{th}_{i,u+1}$ and the sequence in memory will be set to the smaller matching result and the corresponding sequence.

The method presented above takes an incremental way. It need not that all training data are recorded before the training starts. Whenever a demonstration is available, the task models will be updated. It is suitable for the robot learning scenario. Pardowitz et al. [92] also chose an incremental way to learn the Markov model of the task. The difference is that the task models used by us update the weights of the transitions between the primitives during the incremental learning. The TPG proposed by Pardowitz et al., which can be initialized as a general model with one instance, is restricted by dropping unessential precedence relations step by step when a robot sees new demonstrations.

6.3 Extending Task Knowledge

In the section above, a semi-supervised task learning strategy was introduced. Several task models are initialized according to the labeled data and optimized afterwards using unlabeled data, where the number of task models is constant from the beginning. If an unseen task appears, it would be simply rejected by the learning process. However, we aim for a robot which can discover a new task when it has seen it several times, even if this task has not been seen by the robot, before. This requirement is called *novelty detection* in pattern recognition. Detecting novel events is an important ability of many signal classification schemes given the fact that we never can train a machine learning system on all possible object classes with instances the system is likely to encounter. For example, the novelty detection has appeared in fields like visual surveillance [24], topic detection and tracking in news [64], and musical material analysis [97]. Xiang and Gong argue that the online and incremental model construction is not only desirable but also necessary for processing large volumes of unlabeled surveillance video data [129]. They used a bootstrapping dataset to initialize a normal model and an approximated abnormal model with several classes. When a new observation is available, a Likelihood Ratio Test (LRT), which is based on both normal and abnormal models, is used to decide whether it can be accepted by an existing class. If it is accepted, the EM algorithm is used to update the corresponding model. Otherwise, a new abnormal class is generated and the weight renormalization is performed.

In our scenario, there is no abnormal model. Actually, if such a model exists, it is taken as a normal model by us. Therefore, our learning strategy is based on current task models. Because of the rejection mechanism in the semi-supervised learning process, the rejected data is buffered and built into new tasks in an unsupervised way. The process is shown in Figure 6.2. Considering also the possible false negative task rejection, a buffer with a pre-defined length l_b is used to save the rejected sequences. Once the buffer is full, a temporary model is built up based on the sequences in it. If all similarity measurements between the sequences and the new model are above the loosest match threshold Th_o,

/* ***Learning new tasks*** */
insert rejected sequence P_m into buffer
if buffer is full **then**
1. build a task model Λ_{n+1}
2. $k^* = \arg\min_{k=1...l_b}(s(n+1, P_k^{\text{buffer}}))$

if $s(n+1, P_{k^*}^{\text{buffer}}) \geq Th_o$ **then**
1. take Λ_{n+1} as a valid task model
2. $n \leftarrow n+1$

else $\{s(n+1, P_{k^*}^{\text{buffer}}) < Th_o\}$
delete the sequence $P_{k^*}^{\text{buffer}}$

Figure 6.2: Pseudocode of new task learning

this model is taken as a valid learned new task model and the buffer is emptied for a next new model. Otherwise, the sequence with lowest similarity will be deleted from the buffer. Then, the next task rejection refills it and triggers the process again. In fact, this strategy assumes that a new task is demonstrated to the system at least l_b times in order to extend the task knowledge.

6.4 Experiment

To assess the semi-supervised learning of the manipulative tasks, three manipulative tasks are used. They are the same as the tasks used for task recognition in Section 5.5. For every task, the whole training set contains the primitive sequences detected from 20 observations, 16 sequences are testing data. In the first evaluation, we used different lengths of labeled data to start the learning process. The lowest matching threshold Th_0 is 0. The labeled and unlabeled data are randomly chosen from the training set. Figure 6.3 shows recognition rates of the different tasks given different lengths of labeled data in the training set. In the figure, the error rates of the task recognition, especially the error rates

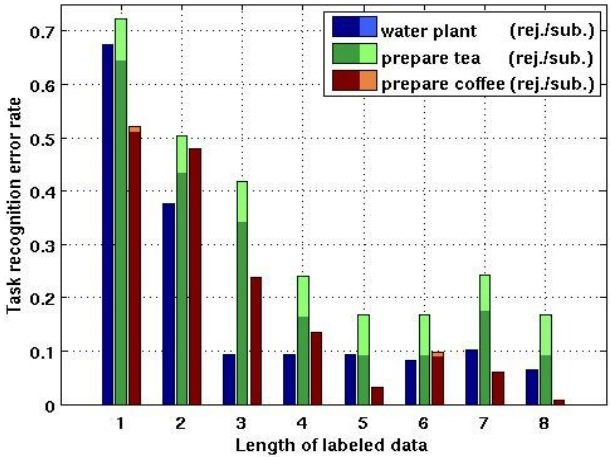


Figure 6.3: The recognition error rates based on different lengths of labeled data

caused by rejection, decrease quickly when the length of labeled data increase from 1 to 4, and become stable when the length gets larger. That indicates the acceptance threshold of the task models can not be correctly set without enough labeled data. The reason for that is that too few labeled sequences of a task could eventually contain primitive detection errors and lead to biased models. In this situation, the threshold of learned models stays quite high and the similarity measurements for the unlabeled sequences are small. Thus, the system rejects them as unseen tasks. In fact, it is the model correctness problem in semi-supervised learning. In the survey of Zhu [137], it says that "If a mixture model assumption is correct, unlabeled data is guaranteed to improve accuracy. However if the model is wrong, unlabeled data may actually hurt accuracy." Although we are not using a mixture of gaussian model, the model correctness is also a key issue here. As shown in the results, too few labeled data caused a false initialization of the task models and the semi-supervised learning did not work after that. The substitution error of task "prepare tea" is higher than the other two because of the insertion error of the manipulative primitives which belong to the task "prepare coffee", which has been explained in Section 5.5.

An often asked question in semisupervised learning is "does unlabeled data always help?" The answer is "no". This phenomenon is already observed by many researchers. To test whether the learned task models are improved by unlabeled data in our learning mechanism, we compare the recognition results between using task models learned from pure supervised learning and semisupervised learning with the same length of labeled data. Figure 6.4 shows the results. The x-axis represents the length of labeled data used in the

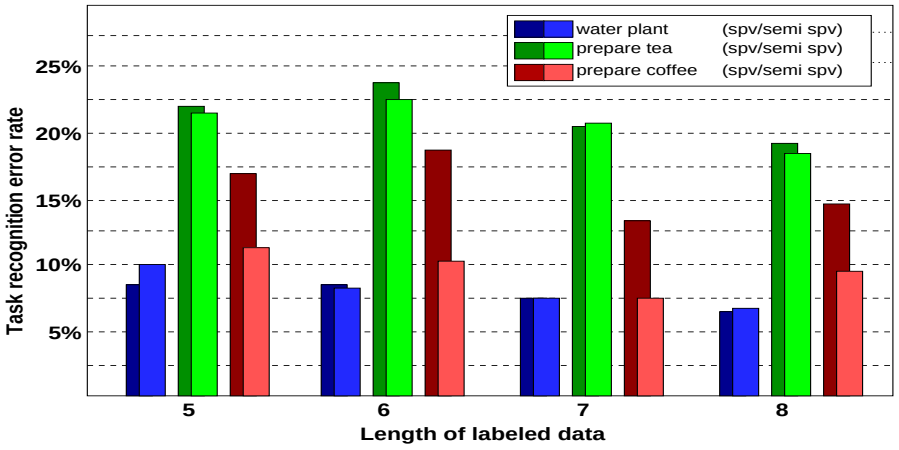


Figure 6.4: The task recognition error rate based on the models learned from pure supervised method and semi-supervised method.

learning process. Because the semi-supervised learning only works when the models are correctly initialized, we started from length of 5. The recognition results from the tasks "water plant" and "prepare tea" have not been heavily effected by the unlabeled data. But the error rate from "prepare coffee" did decrease a lot. Our analysis is that this is because of the complexity of task "prepare coffee". Compared to the other two tasks, it has more flexibility and more possible manipulative primitive sequences. This variance is hard to be learned from a few labeled data. Therefore, we think the semi-supervised learning can

improve the results only when the task is relatively complex and its model can not be well constructed by a small labeled training set.

In the next experiment, only labeled data from two tasks are given in order to test the new task learning strategy. From the results in Figure 6.3 it is found that a valid task model cannot be constructed with less than 5 labeled data items. So the length of labeled observations and the buffer of rejected observations l_b are both set to 5. Figure 6.5 shows the task recognition error rates on the testing set when choosing different initial tasks. The

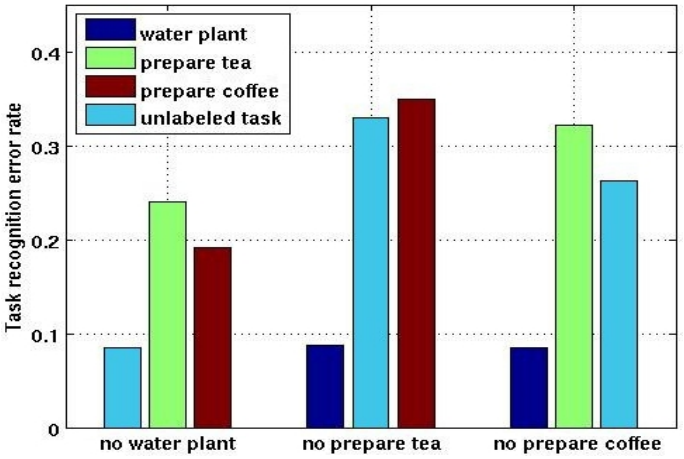


Figure 6.5: The task recognition error rate based on labeled data only from two tasks where the third unlabeled task (light blue) is learned.

results indicate that the unlabeled task was detected and correctly built up. However, a degeneration of recognition performance is also observed, especially when one of the tasks "prepare tea" and "prepare coffee" is not labeled at the beginning. This is because of the false rejections in the learning process. The initial sequences of unlabeled task could also contain errors.

6.5 Summary

Social robots that act in domestic environments need to be able to extend their knowledge both on the learned tasks and unknown tasks during the interaction with their environment. Here we focus on the learning for the recognition of manipulative tasks. A semi-supervised incremental task learning strategy is put forward in this chapter. It aims for the situation that the robot can start the learning by a few labeled training samples and optimize the models afterwards without the need of a human pre-labeling. The task models are based on a Markov process that is coupled to a HMM-based recognition of primitive actions. It has a probabilistic similarity measure of an unlabeled sequence given a learned model, where a random model is taken into consideration to adjust the score of a primitive sequence. Thereby, the system is able to reject unseen tasks. A solution is also presented that collects and filters the rejected tasks in a buffer which is used to learn new models in an unsupervised way. This scheme provides the robot the ability to pre-structure its observation. In future work this could be the basis for an active learning

strategy of the robot that could ask questions about previously unknown task sequences observed.

The experiments show the applicability of this approach. Pre-learned tasks were stably recognized from an initially labeled set of only five samples. Additional tasks that were previously unknown were newly instantiated and successfully recognized on a test set.

7. Summary and Conclusion

With the development of modern robots, they don't only stay in the factories but also join our daily life. They could work in normal indoor environments and communicate with non-expert users. To maintain a natural human-robot interaction, the robot needs to understand the human via different observational modalities. In this dissertation we focus on the vision-based recognition of manipulative gestures. The manipulative gestures are hand actions that are defined by a non-deterministic sequence of object manipulations. They are often considered as the interaction between humans and their environment and being irrelevant to the communication between humans or humans and robots. We argue that the manipulative gestures serve an important communicative function in human-robot interaction. First, the manipulation of an object draws the attention of the communication partner on the objects that are relevant for a performed task. Secondly, it serves the goal of a more pro-active behavior of the robot in passive, more observational situations.

However, most techniques for the recognition of symbolic, interactional or referential gestures cannot be transferred because they ignore the object context and assume an object independent characteristic of the hand trajectory. Other approaches that focus on action recognition either use a pure semantic approach without considering motion models or simplify the trajectory segmentation problem in a pure bottom-up process.

The presented work overcomes several of these deficiencies. Being primarily focused on computer vision our goal is to stay as connected to the visual signal as possible, where predefined actions and particular semantic labels have direct visual correlates. In order to differentiate between similar trajectories, the objects manipulated are taken as context by us. Comparing to the methods which couple the object context in a hand-centered or parallel way, our approach is called object-oriented. It has two different aspects: it is object-centered in terms of trajectory features that are defined relative to an object and it uses object-specific models for manipulative primitives.

There are two commen difficulties in the recognition of action primitives: (i) the segmentation ambiguity and (ii) spatio-temporal variability of the hand trajectory. In addition to these two, we have to cope with an even harder situation, (iii) the recognition can

not be view-dependent. It is because that a mobile robot at home won't always observe the user's object manipulations from the same view point. Human actions happen in a 3-D space, intuitively most research work for the view-invariant recognition of human actions chose 3-D approaches. In our work, we put forward a novel 2-D approach. The reason for this choice has two folds. Firstly, there are still problems in the 3-D tracking in mono-camera images. Better tracking results can be achieved by using stereo cameras. But it poses further constraints on the hardware setting. Secondly, the limited view-angle variance in our scenario and the represention of the actions by the relative movements restricts the variance of the trajectories. For the segmentation ambiguity, we used the vicinities of the contextual objects for a pre-segmentation of the hand trajectory. The vicinities in 2D images are approximated as ellipses and only dependend on the tilt angle of the robot camera and the size of detected objects in the images. For a better modeling of primitives, three kinds of variations of the trajectories are analyzed: the variance of the repetition of a primitive by one performer, the variance of the performance of a primitive from different performers, and the variance of the observation on a primitive performed by a person but from different view-angles. It is found that the latter two variations are hard to be identified when both of them are taken into consideration at the same time. If we simply put them together and model them as random models, it will cause a too general model and brings much insertion errors. Therefore, our system does only a person dependent recognition. The repetition variance of the primitives is taken as random variance and covered by the observational probabilities of HMMs. The variance caused by different view-angles is thought as systematic errors. It is modeled by adding an extra node into the topology of HMM, which nonlinearly scales the mean values of the HMMs in the matching process. Furthermore, a semi-continuous feature vector, which represents the relative movement between hand and objects, was proposed by us. The feature which is less affected by the view-angles are represented continuously. Otherwise it is coded by coarse discrete values. This representation also to certain extent limits the variance of the trajectories from different view-angles in 2D images. In the recognition phase, the manipulative primitives are spotted by a particle filter approach. It matches object specific HMMs in a more flexible way than the traditional forward-backward algorithm due to an explicit modeling of an action abortion and resampling step. It has also more flexible transitions between model states than condensation-based trajectory matching.

Not only focusing on the recognition of manipulative primitives, we also try to achieve higher level understanding of the manipulative gestures, e.g., the manipulative tasks. A two-layered recognition structure was proposed by us. It can be represented by an unified graphical model. The lower layer has HMM for the object-specific manipulative primitives. It is coupled with the task-specific Markovian models on the upper level. Whether a primitive sequence provided by the lower level is accepted as a valid task depends on the similarity measurements between the sequence and the task models. To obtain a comparable similarity measurement between the sequences with different lengths, a random model is used to adjust the score of a primitive sequence given the Markov model of a task. Aiming to have a intuitive task learning of robots, we put forward a semi-supervised incremental learning strategy. Taking a set of pre-learned object-specific manipulative primitives as basic states, the task models can be initialized with a few labeled data, and updated continuously when new unlabeled data is available. With a

threshold for task acceptance, the system is able to reject unseen tasks. Then, a solution is presented by us that collects and filters the rejected tasks in a buffer which is used to learn new models in an unsupervised way. This scheme provides the robot the ability to pre-structure its observation. In future work this could be the basis for an active learning strategy of the robot that could ask questions about previously unknown task sequences observed.

Another problem we need to consider in manipulative gesture recognition is the attention problem. Because of the object-specific primitive definition and the parallel processing for each detected object, the system has to start a lot of processing threads and searches correct primitives in a much bigger data space when there are many objects appearing in the scene simultaneously. It is inefficient and error-prone. Therefore, we proposed an architecture realizing a combined bottom-up and top-down processing loop. The top-down processing utilizes the task-level prediction of possible primitives to restrict the object types possibly detected as well as the action primitives possibly recognized. In the bottom-up path, a processing thread is created for each detected object according to the currently recognized tasks. Afterwards, the primitive detected in the threads are fed back into the task level in order to update the prediction closing the bottom-up and top-down cycle. Thereby, a dynamic attention mechanism is realized that reduces the number of considered objects and simplifies the segmentation task of the hand trajectory.

We were able to show first experiments that underline the potential of the presented approach. A scenario in an office environment was set up. In the first experiments, we evaluated the view-variant manipulative gesture primitive recognition. The images were recorded from different view-angles and different distances. The action primitives were recognized quite robustly. The proposed primitive model showed better performance than HMM. In the task recognition and learning, the top-down attention filter significantly improves the computation time as well as the recognition performance. The results also presented the limitation of the semi-supervised learning that it need a certain length of labeled data to correctly initialize the task model. If this condition is satisfied, the task models can be built up. Furthermore, unseen tasks can also be learned.

Further work needs to concentrate on several issues. In terms of feature description neither pure symbolic nor trajectory-based characterizations will be general enough to describe the huge variety of manipulative actions. Trajectory-based features allow to distinguish actions that do not result in observable state changes of the objects, but suffer from large trajectory variations. The proposed object specific motion-models account for these variations to a certain degree. How to deal with multiple representations on both symbolic and subsymbolic levels is still an open research question. The coupling of motion models and object types also leads to another important aspect of actions: the concept of object affordances. The observed shape and function of an object activates an expected set of hand trajectories and vice versa. We expect that this kind of coupling will be a key issue both in categorization of objects and learning new action verbs. Another aspect is the development of more sophisticated task models that need to include human intentions on multiple scopes of time and space. Finally, more sophisticated experiments are needed to evaluate current action recognition approaches. Appropriate benchmark datasets for manipulative action recognition are currently not available and most approaches focus on their specific application domain. Despite of the open issues, the presented dissertation

has shown the first step to the vision-based manipulative gesture recognition in a human-robot interaction scenario, and is therefore a first attempt to bring robots closer to our daily life.

Bibliography

[1] J. Alon, V. Athitsos, and S. Sclaroff. Accurate and efficient gesture spotting via pruning and subgesture reasoning. In *Proc. ICCV Human-Computer Interaction Workshop*, pages 189–198, 2005.

[2] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear non-gaussian bayesian tracking. *IEEE Trans. Signal Process*, 50(2):174–188, 2002.

[3] D. H. Ballard and Chen Yu. A multimodal learning interface for word acquisition. In *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP'03)*, pages 784–787, 2003.

[4] Ketan Barhate, Kaustubh Patwardhan, Sumantra Dutta Roy, Subhasis Chaudhuri, and Santanu Chaudhury. Robust shape based two hand tracker. In *International Conference on Image Processing*, pages 1017–1020, 2004.

[5] Faisal I. Bashir, Ashfaq A. Khokhar, and Dan Schonfeld. View-invariant motion trajectory-based activity classification and recognition. *Multimedia Syst.*, 12(1):45–54, 2006.

[6] M. J. Black and A. D. Jepson. A probabilistic framework for matching temporal trajectories: Condensation-based recognition of gestures and expressions. In *European Conf. on Computer Vision, ECCV-98*, pages 909–924, Freiburg, Germany, 1998.

[7] A. Bobick. Computers seeing action. In *Proceedings of BMVC*, pages 13–22, 1996.

[8] A. Bobick. Movement, activity, and action: The role of knowledge in the perception of motion. In *Royal Society Workshop on Knowledge-based Vision in Man and Machine*, 1998.

[9] A. F. Bobick and Y. A. Ivanov. Action recognition using probabilistic parsing. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 196, Washington, DC, USA, 1998. IEEE Computer Society.

[10] Aaron F. Bobick and James W. Davis. The recognition of human movement using temporal templates. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(3):257–267, 2001.

[11] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, page 994, Washington, DC, USA, 1997. IEEE Computer Society.

[12] R. A. Brooks. The intelligent room project. In *CT '97: Proceedings of the 2nd International Conference on Cognitive Technology (CT '97)*, page 271, Washington, DC, USA, 1997. IEEE Computer Society.

[13] H. Bui. A General Model for online Probabilistic Recognition. In *Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-2003)*, Acapulco, Mexico, 2003.

[14] H. Bui, D. Phung, and S. Venkatesh. Hierarchical hidden markov models with general state hierarchy. In *AAAI-04*, 2004.

[15] H. H. Bui. Policy recognition in the abstract hidden markov model. *Journal of Artificial Intelligence Research*, 17:451–499, 2002.

[16] John Bulwer. *Chirologia: or the Naturall Language of the Hand*. London, 1644.

[17] L. Campbell, D. Becker, A. Azarbayejani, A. Bobick, and A. Pentland. Invariant features for 3-d gesture recognition. pages 157–162, 1996.

[18] M.T. Chan, A. Hoogs, J. Schmiederer, and M. Petersen. Detecting rare events in video using semantic primitives with hmm. In *ICPR04*, pages IV:150–154, 2004.

[19] Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In Arivind Joshi and Martha Palmer, editors, *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 310–318, San Francisco, 1996. Morgan Kaufmann Publishers.

[20] N. Chomsky. Three models for the description of language. *Information Theory, IEEE Transactions on*, 2(3):113–124, 1956.

[21] Noam Chomsky. *Syntactic structures*. Mouton, 1957.

[22] Herbert H. Clark. *Pointing: where language, culture, and cognition meet: Pointing and Placing*, chapter 10, pages 243–268. Erlbaum, Mahwah, 2003.

[23] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[24] T. Duong, H. Bui, D. Phung, and S. Vekatesh. Activity recognition and abnormality detectiong with the switching hidden semi-markov model. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2005.

[25] Stefan Eickeler, Andreas Kosmala, and Gerhard Rigoll. Hidden markov model based continuous online gesture recognition. In *Proceedings of the 14th International Conference on Pattern Recognition*, pages 1206–1208, Washington, DC, USA, 1998. IEEE Computer Society.

[26] S. Ekvall, D. Aarno, and D. Kragic. Task learning using graphical programming and human demonstrations. In *The 15th IEEE International Symposium on Robot and Human Interactive Communication (ROMAN 2006)*, pages 398 – 403, 2006.

[27] Alan Fern, Jeffrey Mark Siskind, and Robert Givan. Learning temporal, relational, force-dynamic event definitions from video. In *Eighteenth national conference on Artificial intelligence*, pages 159–166, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[28] S. Fine, Y. Singer, and N. Tishby. The hierarchical hidden markov model: Analysis and applications. *Machine Learning*, 32(1):41–62, 1998.

[29] Michael Fleischman, Phillip Decamp, and Deb Roy. Mining temporal patterns of movement for video content classification. In *MIR '06: Proceedings of the 8th ACM international workshop on Multimedia information retrieval*, pages 183–192, New York, NY, USA, 2006. ACM Press.

[30] Michael Fleischman and Deb Roy. Why verbs are harder to learn than nouns: Initial insights from a computational model of intention recognition in situated word learning. In *Proc. of the 27th Annual Meeting of the Cognitive Science Society*, 2005.

[31] T. Fong, I. Nourbakhsh, and K. Dautenhahn. A survey of socially interactive robots. *Robotics and Autonomous Systems*, 42, 2002.

[32] David A. Forsyth and Jean Ponce. *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference, 2002.

[33] J. Fritsch. *Vision-based Recognition of Gestures with Context*. Dissertation, Bielefeld University, 2003.

[34] J. Fritsch, N. Hofemann, and G. Sagerer. Combining Sensory and Symbolic Data for Manipulative Gesture Recognition. In *Proc. IEEE ICPR*, pages 930–933, Cambridge, UK, 2004.

[35] J. Fritsch, M. Kleinehagenbrock, A. Haasch, S. Wrede, and G. Sagerer. A flexible infrastructure for the development of a robot companion with extensible HRI-capabilities. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 3419–3425, Barcelona, Spain, April 2005.

[36] J. Fritsch, F. Loemker, M. Wienecke, and G. Sagerer. Detecting assembly actions by scene observation. In *Proceedings International Conference on Image Processing*, volume I, pages 212–215, Vancouver, Sep. 2000. IEEE.

[37] T. Fukuda, Y. Nakauchi, K. Noguchi, and T. Matsubara. Time series action support by mobile robot in intelligent environment. In *Proc. IEEE Int'l Conf. Robotics and Automation*, pages 2908–2913, Barcelona, Spain, 2005.

[38] Aphrodite Galata, Neil Johnson, and David Hogg. Learning variable-length markov models of behavior. *Comput. Vis. Image Underst.*, 81(3):398–413, 2001.

[39] Jiang Gao, Alexander G. Hauptmann, Ashok Bharucha, and Howard D. Wactlar. Dining activity analysis using a hidden markov model. In *ICPR '04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 2*, pages 915–918, Washington, DC, USA, 2004. IEEE Computer Society.

[40] D. M. Gavrila. The visual analysis of human movement: a survey. *Comput. Vis. Image Underst.*, 73(1):82–98, January 1999.

[41] G. Gergely. What should a robot learn from an infant? mechanisms of action interpretation and observational learning in infancy. In *Proceedings Third International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*, pages 13–24, Boston, MA, USA., 2003.

[42] Axel Grossmann, Matthias Wendt, and Jeremy Wyatt. A semi-supervised method for learning the structure of robot environment interactions. In *IDA*, pages 36–47, 2003.

[43] Dick Grune and Ceriel J. H. Jacobs. *Parsing techniques: a practical guide.* Ellis Horwood, Upper Saddle River, NJ, USA, 1990.

[44] A. Haasch, S. Hohenner, S. Huwel, M. Kleinehagenbrock, S. Lang, I. Toptsis, G. A. Fink, J. Fritsch, B. Wrede, and G. Sagerer. Biron — the bielefeld robot companion. In *Proc. Int. Workshop on Advances in Service Robotics*, pages 27–32, Stuttgart, Germany, 2004.

[45] Jesse Hoey, Axel von Bertoldi, Pascal Poupart, and Alex Mihailidis. Assisting persons with dementia during handwashing using a partially observable markov decision process. In *ICVS 2007, 5th International Conference on Computer Vision Systems*, Bielefeld University, 2007.

[46] N. Hofemann, J. Fritsch, and G. Sagerer. Recognition of deictic gestures with context. volume 3175 of *Lecture Notes in Computer Science*, pages 334–341, Heidelberg, Germany, 2004. Springer-Verlag.

[47] Nils Hofemann. *Videobasierte Handlungserkennung fuer die natuerliche Mensch-Maschine-Interaktion.* Dissertation, Bielefeld University, 2007.

[48] P. Hong, M. Turk, and T. Huang. Constructing finite state machines for fast gesture recognition. In *Proceedings of the International Conference on Pattern Recognition*, page 3695, Washington, DC, USA, 2000. IEEE Computer Society.

[49] Somboon Hongeng, Ram Nevatia, and Francois Bremond. Video-based event recognition: activity representation and probabilistic recognition methods. *Comput. Vis. Image Underst.*, 96(2):129–162, 2004.

[50] Xuedong Huang, Yasuo Ariki, and Mervyn Jack. *Hidden Markov Models for Speech Recognition.* Columbia University Press, New York, NY, USA, 1990.

[51] Stephen S. Intille and Aaron F. Bobick. Recognizing planned multiperson action. *Comput. Vis. Image Underst.*, 81(3):414–445, March 2001.

[52] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. *Lecture Notes in Computer Science*, 1064:343–356, 1996.

[53] Michael Isard and Andrew Blake. A mixed-state condensation tracker with automatic model-switching. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, page 107, Washington, DC, USA, 1998. IEEE Computer Society.

[54] Yuri A. Ivanov and Aaron F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):852–872, 2000.

[55] K.H. Jo, Y. Kuno, and Y. Shirai. Manipulative hand gesture recognition using task knowledge for human computer interaction. In *Proc. Int'l Conf. on Automatic Face and Gesture Recognition*, pages 468–473, 1998.

[56] G. Johansson. Visual perception of biological motion and a model for its analysis. *Percept. Phychophys*, 14:201–211, 1973.

[57] Adam Kendon. *Studies in Dyadic Communication*, chapter Some relationships between body motion and speech, pages 177–216. Pergamon Press, 1972.

[58] G. Drew Kessler, Larry F. Hodges, and Neff Walker. Evaluation of the cyberglove as a whole-hand input device. *ACM Trans. Comput.-Hum. Interact.*, 2(4):263–283, 1995.

[59] Bassam Khadhouri and Yiannis Demiris. Compound effects of top-down and bottom-up influences on visual attention during action recognition. In *IJCAI*, pages 1458–1463, 2005.

[60] R. Kjeldsen and J.R. Kender. Visual hand gesture recognition for window system control. In *Proc. Int'l Workshop Automatic Face and Gesture Recognition*, pages 184–188, 1995.

[61] Marcus Kleinehagenbrock. *Interaktive Verhaltenssteuerung fuer Robot Companions*. phdthesis, Universitaet Bielefeld, Technische Fakultaet, 2005.

[62] E. Kurtenbach, G. & Hulteen. *The Art and Science of Interface Design*, chapter Gestures in Human Computer Communication, pages 309–317. Addison-Wesley publisher, 1990.

[63] Sebastian Lang, Marcus Kleinehagenbrock, Sascha Hohenner, Jannik Fritsch, Gernot A. Fink, and Gerhard Sagerer. Providing the basis for human-robot-interaction: a multi-modal attention system for a mobile robot. In *ICMI '03: Proceedings of the 5th international conference on Multimodal interfaces*, pages 28–35, New York, NY, USA, 2003. ACM Press.

[64] Victor Lavrenko, James Allan, Edward DeGuzman, Daniel LaFlamme, Veera Pollard, and Stephen Thomas. Relevance models for topic detection and tracking. In

Proceedings of the second international conference on Human Language Technology Research, pages 115–121, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[65] Hyeon-Kyu Lee and Jin H. Kim. An HMM-based threshold model approach for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):961–973, 1999.

[66] S. Li, M. Kleinehagenbrock, J. Fritsch, B. Wrede, and G. Sagerer. "BIRON, let me show you something": Evaluating the interaction with a robot companion. In *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics, Special Session on Human-Robot Interaction*, pages 2827–2834, The Hague, The Netherlands, October 2004. IEEE.

[67] Zhe Li, Jannik Fritsch, Sven Wachsmuth, and Gerhard Sagerer. An object-oriented approach using a top-down and bottom-up process for manipulative action recognition. In *Annual Symposium of the German Association for Pattern Recognition (DAGM)*, pages 212–221, Berlin, Germany, 2006. Springer-Verlag.

[68] Zhe Li, Nils Hofemann, Jannik Fritsch, and Gerhard Sagerer. Hierarchical modeling and recognition of manipulative gesture. In *Proc. ICCV, Workshop on Modeling People and Human Interaction*, Beijing, China, 2005. IEEE.

[69] Zhe Li, Sven Wachsmuth, Jannik Fritsch, and Gerhard Sagerer. *Manipulative Action Recognition for Human-Robot Interaction*, incollection Vision Systems: Segmentation and Pattern Recognition. I-Tech Education and Publishing, Vienna, Austria EU, 2007.

[70] Zhe Li, Sven Wachsmuth, Jannik Fritsch, and Gerhard Sagerer. Semi-supervised incremental learning of manipulative tasks. In *Proc. of IAPR Conference on Machine Vision Applications (MVA 2007)*, pages 73–77, Tokyo, Japan, May 2007.

[71] Zhe Li, Sven Wachsmuth, Jannik Fritsch, and Gerhard Sagerer. View-adaptive manipulative action recognition for robot companions. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, San Diego, CA, USA, October 2007. IEEE.

[72] L. Lin, D. Fox, and H. Kautz. Learning and Inferring Transportation Routine. In *Proceedings of AAAI-04*, San Jose, California, 2004.

[73] Long-Ji Lin. *Reinforcement learning for robots using neural networks*. PhD thesis, Pittsburgh, PA, USA, 1992.

[74] Nianjun Liu, Brian C. Lovell, Peter J. Kootsookos, and Richard I. A. Davis. Model structure selection and training algorithms for an hmm gesture recognition system. In *IWFHR '04: Proceedings of the Ninth International Workshop on Frontiers in Handwriting Recognition*, pages 100–105, Washington, DC, USA, 2004. IEEE Computer Society.

[75] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 20:91–110, 2003.

[76] S. J. McKenna and S. Gong. Gesture recognition for visually mediated interaction using probabilistic event trajectories. In *The British Machine Vision Conference (BMVC1998)*, Southampton, United Kingdom, 1998.

[77] T. Michalke, M. Schneider, A. Gepperth, J. Fritsch, and C. Goerick. Towards a human-like vision system for resource-constrained intelligent cars. In *ICVS 2007, 5th International Conference on Computer Vision Systems*, Bielefeld University, 2007.

[78] D. Minnen, I. Essa, and T. Starner. Expectation grammars: leveraging high-level expectations for activity recognition. In *Proc. Computer Vision and Pattern Recognition*, volume 2, pages 626–632, June 2003.

[79] Darnell Moore and Irfan Essa. Recognizing multitasked activities from video using stochastic context-free grammar. In *Eighteenth national conference on Artificial intelligence*, pages 770–776, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[80] D.J. Moore, I.A. Essa, and M.H. Hayes, III. Exploiting human actions and object context for recognition tasks. In *Proc. ICCV*, pages 20–27, 1999.

[81] P. Morguet and M. Lang. Spotting dynamic hand gestures in video image sequences using hidden markov models. In *International Conference on Image Processing*, pages 193–197, Chicago, USA, 1998.

[82] K. P. Murphy and Mark A. Paskin. Linear-time inference in hierarchical hmms. In *Proc. of Neural Information Processing Systems*, pages 833–840, 2001.

[83] Kevin Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, UC Berkeley, Computer Science Division, 2002.

[84] C. L. Nehaniv, K. Dautenhahn, J. Kubacki, M. Haegele, C. Parlitz, and R. Alami. A methodological approach relating the classification of gesture to identification of human intent in the context of human-robot interaction. In *Proc. IEEE Ro-man 05*, Nashville, USA, 2005.

[85] C. P. Nehaniv. Classifying types of gesture and inferring intent. In *Proceedings of the Symposium on Robot Companions: Hard problems and Open Challenges in Robot-Human Interaction AISB'05*, pages 74–81, Hatfield, UK, 2005.

[86] N. Nguyen, D. Phung, S. Venkatesh, and H. Bui. Learning and detecting activities from movement trajectories using the hierarchical hidden markov model. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2005.

[87] Koichi Ogawara, Jun Takamatsu, Hiroshi Kimura, and Katsushi Ikeuchi. Generation of a task model by integrating multiple observations of human demonstrations. In *ICRA*, pages 1545–1550, 2002.

[88] Koichi Ogawara, Jun Takamatsu, Hiroshi Kimura, and Katsushi Ikeuchi. Modeling manipulation interactions by hidden markov models. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1096–1101, 2002.

[89] Ryuichi Oka. Spotting method for classification of real world data. *The Computer Journal*, 41(8):559–565, 1998.

[90] N. Oliver, A. Garg, and E. Horvitz. Layered representations for learning and inferring office activity from multiple sensory channels. *Computer Vision and Image Understanding*, 96(2):163–180, 2004.

[91] Nuria Oliver, Barbara Rosario, and Alex Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, 2000.

[92] M. Pardowitz, R. Zöllner, and R. Dillmann. Unsupervised and incremental acquisition of and reasoning on holistic task knowledge for household robot companions. In *IROS06*, Beijing, China, 2006.

[93] Vladimir Pavlovic, Rajeev Sharma, and Thomas S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):677–695, 1997.

[94] D. I. Perret, M Harries, R. Bevan, S. Thomas, P.J.Benson, A.J. Mistlin, A.J. Chitty, J.K. Hietanene, and J.E. Ortega. Frameworks of analysis for the neural representation of animate objects and actions. *Journal of Experimental Biology*, 146:87–113, 1989.

[95] Thies Pfeiffer and Marc Erich Latoschik. Interactive Social Displays. In *Proceedings of the IEEE Symposium on 3D User Interfaces 2007*, North Carolina, USA, 2007.

[96] C. S. Pinhanez and A. F. Bobick. Human action detection using pnf propagation of temporal constraints. In *Proc. IEEE CVPR*, pages 898–907, Washington, DC, USA, 1998.

[97] Thomas Ploetz, Gernot A. Fink, Peter Husemann, Sven Kanies, Kai Lienemann, Tobias Marschall, Marcel Martin, Lars Schillingmann, Matthias Steinruecken, and Henner Sudek. Automatic detection of song changes in music mixes using stochastic models. In *Proc. of the Int. Conf. on Pattern Recognition*, volume 3, pages 665–668. IEEE, 2006.

[98] R. Polana and R. Nelson. Low level recognition of human motion (or how to get your man without finding his body parts. In *IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pages 77–82, 1994.

[99] David V. Pynadath and Michael P. Wellman. Probabilistic state-dependent grammars for plan recognition. In *Proceedings of the 16th Annual Conference on Uncertainty in Artificial Intelligence*, pages 507–514, 2000.

[100] Francis Quek, David McNeill, Robert Bryll, Susan Duncan, Xin-Feng Ma, Cemil Kirbas, Karl E. McCullough, and Rashid Ansari. Multimodal human discourse: gesture and speech. *ACM Trans. Comput.-Hum. Interact.*, 9(3):171–193, 2002.

[101] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Readings in speech recognition*, pages 267–296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[102] Cen Rao, Alper Yilmaz, and Mubarak Shah. View-invariant representation and recognition of actions. *Int. J. Comput. Vision*, 50(2):203–226, 2002.

[103] Sira Panduranga Rao and Diane J. Cook. Predicting inhabitant action using action and task models with application to smart homes. *International Journal on Artificial Intelligence Tools*, 13(1):81–99, 2004.

[104] G. Rizzolati, L. Fadiga, V. Gallese, and L. Fogassi. Premotor cortex and the recognition of motor actions. *Cognitive Brain Research*, 3:131–141, 1996.

[105] Neil Robertson and Ian Reid. Behaviour understanding in video: A combined method. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pages 808–815, Washington, DC, USA, 2005. IEEE Computer Society.

[106] Myung-Cheol Roh, Ho-Kuen Shin, Sang-Woong Lee, and Seong-Whan Lee. Volume motion template for view-invariant gesture recognition. In *International Conference on Pattern Recognition (ICPR)*, pages 1229–1232, Hong Kong, China, 2006.

[107] Sam T. Roweis and Zoubin Ghahramani. A unifying review of linear gaussian models. *Neural Computation*, 11(2):305–345, 1999.

[108] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition edition, 2003.

[109] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, Feb. 1978.

[110] S. Salvador and P. Chan. Fastdtw: Toward accurate dynamic time warping in linear time and space. In *Proc. KDD Workshop on Mining Temporal and Sequential Data*, 2004.

[111] S. Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6):233–242, 1999.

[112] J. Schmidt, B. Kwolek, and J. Fritsch. Kernel Particle Filter for Real-Time 3D Body Tracking in Monocular Color Images. In *Proc. of Automatic Face and Gesture Recognition*, pages 567–572, Southampton, UK, April 2006. IEEE.

[113] M. Soriano, S. Huovinen, B. Martinkauppi, and M. Laaksonen. Skin detection in video under changing illumination conditions. In *Proceedings of the International Conference on Pattern Recognition*, pages 839–842, Washington, DC, USA, 2000. IEEE Computer Society.

[114] T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. In *IEEE International Symposium on Computer Vision*, pages 265–270, 1995.

[115] T.E. Starner and A.P. Pentland. Visual recognition of american sign language using hidden markov models. In *International Workshop on Automatic Face and Gesture Recognition*, pages 189–194, Zurich, Switzerland, 1995.

[116] Moritz Störring, Hans Jørgen Andersen, and Erik Granum. Physics-based modelling of human skin colour under mixed illuminants. *Robotics and Autonomous Systems*, 35(3-4):131–142, 2001.

[117] Jun Takamatsu, Koichi Ogawara, Hiroshi Kimura, and Katsushi Ikeuchi. Recognizing assembly tasks through human demonstration. *Int. J. Rob. Res.*, 26(7):641–659, 2007.

[118] John K. Tsotsos. Motion understanding: Task-directed attention and representations that link perception with action. *Int. J. Comput. Vision*, 45(3):265–280, 2001.

[119] Matthew Turk. *Frontiers of Human-Centred Computing, Online Communities and Virtual Environments*, chapter Gesture Recognition. Springer-Verlag, 2001.

[120] Ilkay Ulusoy and Christopher M. Bishop. Generative versus discriminative methods for object recognition. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 258–265, Washington, DC, USA, 2005. IEEE Computer Society.

[121] P. Viola and M. Jones. Robust real-time object detection. In *Proc. IEEE Int. Workshop on Statistical and Computational Theories of Vision*, Canada, 2001.

[122] Sven Wachsmuth and Gerhard Sagerer. Bayesian networks for speech and image integration. In *Eighteenth national conference on Artificial intelligence*, pages 300–306, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[123] Stefan Waldherr, Roseli Romero, and Sebastian Thrun. A gesture based interface for human-robot interaction. *Auton. Robots*, 9(2):151–173, 2000.

[124] Stefan Waldherr, Sebastian Thrun, Roseli Romero, and Dimitris Margaritis. Template-based recognition of pose and motion gestures on a mobile robot. In *AAAI '98/IAAI '98: Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, pages 977–982, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence.

[125] Daniel Weinland, Remi Ronfard, and Edmond Boyer. Free viewpoint action recognition using motion history volumes. *Comput. Vis. Image Underst.*, 104(2):249–257, November 2006.

[126] M. Wienecke, G. A. Fink, and G. Sagerer. A handwriting recognition system based on visual input. In *2nd International Workshop on Computer Vision Systems*, pages 63–72, Vancouver, Canada, 2001. IEEE.

[127] Andrew D. Wilson and Aaron F. Bobick. Hidden markov models for modeling and recognizing gesture under variation. pages 123–160, 2002.

[128] B. Wrede, A. Haasch, N. Hofemann, S. Hohenner, S. Hüwel, M. Kleinehagenbrock, S. Lang, S. Li, I. Toptsis, G. A. Fink, J. Fritsch, and G. Sagerer. Research issues for designing robot companions: BIRON as a case study. In P. Drews, editor, *Proc. IEEE Conf. on Mechatronics & Robotics*, volume 4, pages 1491–1496, Aachen, Germany, September 2004. Eysoldt-Verlag, Aachen.

[129] T. Xiang and S. Gong. Incremental visual behaviour modelling. In *IEEE International Workshop on Visual Surveillance*, pages 65–72. Graz, Austria, May 2006.

[130] Tao Xiang and Shaogang Gong. Beyond tracking: Modelling activity and understanding behaviour. *Int. J. Comput. Vision*, 67(1):21–51, 2006.

[131] Jinghui Xiao, Bingquan Liu, and Xiaolong Wang. An empirical study of non-stationary ngram model and its smoothing techniques. *Computational LInguistics and Chinese Language Processing*, 12(2):127–154, June 2007.

[132] M. Yamamoto, H. Mitomi, F. Fujiwara, and T. Sato. Bayesian classification of task-oriented actions based on stochastic context-free grammar. In *Proceedings of FGR06*, pages 317–323, 2006.

[133] Guangqi Ye, Jason J. Corso, and Gregory D. Hager. Gesture recognition using 3d appearance and motion features. In *CVPRW '04: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 10*, page 160, Washington, DC, USA, 2004. IEEE Computer Society.

[134] Jie Yin, Xiaoyong Chai, and Qiang Yang. High-level goal recognition in a wireless lan. In *AAAI*, pages 578–584, 2004.

[135] C. Yu and D. H. Ballard. Learning to Recognize Human Action Sequences. In *2nd International Conference on Development and Learning (ICDL'02)*, pages 28–34, 2002.

[136] D. Zhang, D. Gatica-Perez, S. Bengio, and I. McCowan. Modeling individual and group actions in meetings with layered-hmms. *IEEE Transactions on Multimedia*, 8(3):509–520, 2006.

[137] X. Zhu. Semi-supervised learning literature survey. Technical report, University. of Wisconsin Madison, 2 2006. Computer Science TR 1530.

[138] Raoul Zöllner, Michael Pardowitz, Steffen Knoop, and Rüdiger Dillmann. Towards cognitive robots: Building hierarchical task representations of manipulations from human demonstration. In *Proc. of International Conference on Robotics and Automation (ICRA)*, pages 1535–1540, 2005.

Index