Universität Bielefeld
Neuroinformatics Group, Bielefeld University

# Structured Manifolds for Motion Production and Segmentation

## A Structured Kernel Regression Approach

**Jan-F. Steffen**

PhD Thesis

Universität Bielefeld Neuroinformatics Group, Bielefeld University

# Structured Manifolds for Motion Production and Segmentation

## A Structured Kernel Regression Approach

Der Technischen Fakultät der Universität Bielefeld
vorgelegt von

**Jan-F. Steffen**

zur Erlangung des akademischen Grades

**Doktor der Ingenieurwissenschaften**.

Erster Gutachter: Prof. Dr. Helge Ritter (Universität Bielefeld)
Zweiter Gutachter: Prof. Dr. Sethu Vijayakumar (Universität Edinburgh)

Januar 2010

## Acknowledgements

The work presented in this thesis has been carried out within the Neuroinformatics Group, headed by Prof. Dr. Helge Ritter, at the Faculty of Technology, Bielefeld University. It was supported by the German Collaborative Research Centre "SFB 673 - Alignment in Communication" granted by the DFG, and the German Cluster of Excellence 277 "Cognitive Interaction Technology (CITEC)".

In addition, there are a number of people whom I wish to thank for their support, help, time, and the trust that they have put in me and my work.

I would like to thank Helge Ritter, my principal advisor and the first reviewer of this thesis, for all the support and advice he has given me over the past few years. His vision of cognitive robotics and manual intelligence have always been and will always be inspiring to me. I would like to thank him for providing all the possibilities he offered, everything from support of my research stays to labs full of state-of-the-art hardware.

I would like to thank Michael Pardowitz, my office mate for the last two and a half years, who always provided me with valuable advice and became a sort of a backup advisor in many situations. He also helped editing parts of this thesis. His joining the Neuroinformatics Group and "my" office was a veritable piece of luck for me.

I would like to thank Stefan Klanke for introducing me to UKR when I had just started my work and was looking for an interesting topic for my research. Later, during my research stay in Edinburgh, he helped me with everything, ranging from his invaluable support with my work to an introduction to the diversity of the Scottish beers. Last but not least, he read a good part of this thesis and helped me to improve it.

I would like to thank Sethu Vijayakumar for inviting me for a short research stay in the summer of 2008 with his group, the Statistical Machine Learning and Motor Control Group in the Institute of Perception, Action & Behavior at the University of Edinburgh. His friendly nature made me feel like a full member of the group. I would also like to thank him for agreeing to be the second reviewer of this thesis.

I would like to thank Erhan Oztop for inviting me for a short research stay in the summer of 2009 to the Department of Communication and Cognitive Cybernetics (formerly Humanoid Robotics and Computational Neuroscience Department) at the Computational Neuroscience Laboratories of the Advanced Telecommunications Research Institute International (ATR) in Kyoto. In addition to the support with the robots, he did a great job of introducing me to Japan, Japanese Food and even to the world of fishing!

I would like to thank Robert Haschke for his general support and especially for his help with everything concerning Neo/NST, XCF, Vortex, and their many wrappers and wrappers of their wrappers. I would not have had an overview without him.

I would also like to thank all the people in the Neuroinformatics Group, in particular Risto Kõiva for the help with the CyberGlove, Jonathan Maycock for editing parts of this manuscript, Oliver Lieske, Julia Herold and all other "members of the coffee club" for all the nice coffee breaks (and especially Olli for maintaining the coffee maker!). All these people made working in this group such a pleasurable experience.

I would like to thank my girlfriend, Heike Rische, for giving me tremendous support throughout this entire process. It would have been much harder without her. Also, I would like to thank my parents for all the support they have given me throughout my entire life.

# Contents

# 1. Introduction

## 1.1. Motivation

In the recent years, research and engineering have given us a variety of sophisticated robot platforms. Humanoid robots exist whose physical capabilities and movement possibilities have almost attained the complexity of human bodies. Along with its complex body, a large number of sensors for tactile, visual, kinematics and dynamics information retrieval are available to enable the robot to perceive and interact with its environment. In order to process the vast amount of sensory input and to control the motions of complex robots, such systems are endowed with enormous computational power and memory capacities that nearly mimic those of small brains.

However, even if the physical and computational capabilities of robots developed to a sufficient level where one could readily compare robots and humans in these domains, the cognitive abilities of humans are still very much out of reach of even the most sophisticated robots.

In order to understand the core principles of cognition, Ritter et al. (2009) identified "Manual Intelligence", i.e. the ability to dexterously manipulate the environment, as a transition technology that leads from mere movement control to brain-like behaviour. They argue that understanding the generality and flexibility of the human hand and its dextrous control by the human brain will pave the way from simple control to "cognitive interaction".

Driven by this idea, Ritter and his colleagues in Bielefeld have been seeking new insights into hand control and dextrous manipulation for many years. The work presented in this thesis is also inspired by the same far reaching goal of cognitive robots that consciously act as intelligent agents in their environment.

The starting point for this thesis was based on the early work of the author in the domain of dexterous grasping, which led him to the idea of low-dimensional representations of grasping knowledge (Steffen, 2005; Steffen et al., 2007a, 2007b).

Working towards dextrous manipulation, manifolds similar to the grasping case have been developed that are additionally equipped with an enforced intrinsic structure, which inherently represents the underlying manipulation (Steffen et al., 2008b, 2008a; Steffen, Klanke, et al., 2009a, 2009b). Whereas this approach was originally developed to enable a robot to reproduce a demonstrated manipulation movement and to synthesise new motions of the same kind, it constitutes a more general method for the structured representation of sensory data.

Separate to the specific domain of manipulation, the special characteristics of manifolds led to an approach that exploits the intrinsic manifold structure in order to recognise the represented motion from a sequence of observed sensory data. Using a set of several candidate manifolds, the method is also able to segment a sequence consisting of multiple candidate motions into subsequences (Steffen, Pardowitz, & Ritter, 2009a, 2009b; Steffen et al., 2010).

In the course of the following chapters, the concepts and ideas that have been developed,

from the initial grasping approach and the manipulation production up to the recognition and segmentation system will be presented in detail and evaluated according to their usefulness in targeted applications.

## 1.2. Structure of the thesis and contributions

The coarse structure of this thesis is as follows: Chapter 2 gives an overview of a set of existing methods that are fundamental for the entire thesis. Chapters 3 to 6 then present the methods and applications that were developed by the author. Since these chapters present work from different fields of research, each chapter contains a brief introduction to the relevant field along with an overview of related work.

In more detail, the remainder of this thesis is organised as follows:

In **Chapter 2**, a selection of methods is presented that have recently been used in the domain of robotics in the context of motion representation and which are generally qualified to serve as a basis for the ideas of this thesis. In particular, Unsupervised Kernel Regression is reviewed in detail, as this method constitutes the fundamental basis of the main parts of this thesis.

In **Chapter 3**, the acquisition of appropriate hand posture data for complex anthropomorphic robotic hands is detailed. After a brief review of robotic hand models, the core of this chapter is concerned with data acquisition using a data glove along with a new method to map the data glove sensory readings onto a human hand model and then onto a specific robotic hand. This mapping is an important part of this work, since it made the acquisition of manipulation data possible. Prior to this, no sufficiently accurate interface for the recording of sequences of hand posture data had been available in the Neuroinformatics Group at the Bielefeld University.

The data glove mapping was solely carried out by the author.

**Chapter 4** discusses the fundamental ideas and core concepts, which paved the way for the work presented here. In particular, a manifold representation for dextrous grasping is detailed, which gave rise to the concept of *Manipulation Manifolds* and then to the more general *Structured Manifolds*. The work presented here represents the main motivation behind the subsequent work carried out in the field of motion representation.

The manifold representation for grasping, the corresponding algorithm and ideas which lead to *Structured Manifolds*, along with all related issues is the sole work of the author. The basic principles of the *Grasping Manifold* had already been developed during the author's master thesis (German 'Diplomarbeit') (Steffen, 2005), but have been refined and extended in this chapter.

Parts of this chapter have already been published (Steffen, 2005; Steffen et al., 2007a, 2007b, 2008b).

**Chapter 5** introduces the reader to the fundamental ideas behind *Structured Manifolds* for dextrous manipulation. The chapter focusses on representing manipulation data in highly structured manifolds for the purpose of reproducing the represented motions. In addition to this, three motion control algorithms are presented that exploit the intrinsic manifold structure for the motion generation. The most sophisticated one enables the implementation of a closed-loop manipulation controller that can swap Chinese health balls with the Gifu Hand III robotic hand.

The work detailed in Section 5.3 is partly based on the work developed by the author in collaboration with Dr. Stefan Klanke during a research stay in 2008 at the School of Informatics, University of Edinburgh, UK, in Prof. Dr. Sethu Vijayakumar's group. The principal ideas and a major part of the implementation was contributed by the author, but Dr. Stefan Klanke also contributed significantly to the modification of the Matlab UKR toolbox, especially to its extension to periodic latent dimensions, where he also contributed the mathematical basis for the author's idea.

The work presented in Section 5.6 is based on the work developed by the author in collaboration with Dr. Erhan Oztop during a research stay in 2009 at the Computational Neuroscience Laboratories at the Advanced Telecommunications Research Institute International (ATR), Kyoto, Japan, in Prof. Dr. Kenji Doya's group. The principal ideas and the major part of the implementation was contributed by the author.

All other parts of this chapter are solely the work of the author.

Parts of this chapter have already been published (Steffen et al., 2008b, 2008a; Steffen, Klanke, et al., 2009a, 2009b).

**Chapter 6** follows a different direction to the one presented in Chapter 5. Instead of using observed motion data in order to represent the underlying motion, such representations are now exploited for the purpose of the recognition of motion observations. To this end, features in the manifold domain are defined that express the degree of compatibility of observed and represented motions. In addition to a purely manifold-based approach, a more advanced method is incorporated, using competitive neural dynamics for sensory feature grouping for the segmentation of observed data according to a set of candidate motions.

The general idea for the recognition system on the basis of structured manifolds, the UKR-only approach (Section 6.2) as well as the extension to its combination with competitive neural dynamics (Section 6.3.2 to 6.3.10) were contributed by the author.

Parts of this chapter have already been published (Steffen, Pardowitz, & Ritter, 2009a, 2009b; Steffen et al., 2010).

Finally, **Chapter 7** concludes the thesis and provides an outlook for possible future work.

# 2. Related Methods

This chapter presents a selection of methods that are generally qualified to serve as a basis for the ideas presented later on in this thesis.

In particular, *Unsupervised Kernel Regression* is described in detail in the following section, since it constitutes the method of choice for the remainder of the thesis. Subsequently, the *Gaussian Mixture Model* and the *Gaussian Process Latent Variable Model* are briefly summarised as potential alternatives.

## 2.1. Unsupervised Kernel Regression (UKR)

*Unsupervised Kernel Regression* (UKR) is a recent approach to learning non-linear continuous manifold representations of datasets. It was introduced by Meinicke et al. (2005) and developed on the basis of the Generalised Regression Framework (GRF) for unsupervised learning presented in (Meinicke, 2000). The basic idea of the GRF is to reformulate an existing supervised function learning approach in an unsupervised manner. This change of paradigm is realised by only considering training samples of the function *output* instead of supervised *input/output pairs* of data. The data formerly provided by the input then are treated as parameters of the function which are subject to an optimisation procedure during the training.

The following overview of UKR is a brief summary of the original work presented by Meinicke et al. (2005) as well as the further development of the method presented in (Klanke & Ritter, 2006, 2007) and (Klanke, 2007). The summary focusses on the parts of UKR that are important for the remainder of this thesis.

### 2.1.1. The Nadaraya-Watson kernel regression estimator

In the case of UKR, the Nadaraya-Watson kernel regression estimator (Nadaraya, 1964; Watson, 1964) plays the role of the aforementioned supervised approach. In brief, in its original formulation, the estimator uses observed samples $(\mathbf{x}_i, \mathbf{y}_i) \in \boldsymbol{\mathcal{X}} \times \boldsymbol{\mathcal{Y}}$ from the input/output sets $\boldsymbol{\mathcal{X}} \in \mathbb{R}^q$ and $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^d$ (or more precisely, from the joint distribution of two multivariate random variables $X$ and $Y$) in order to derive a corresponding regression function $\mathbf{f}\colon \mathbb{R}^q \to \mathbb{R}^d$:

$$\mathbf{f}(\mathbf{x}; \boldsymbol{\beta}) = \sum_{i=1}^{N} \mathbf{y}_i \frac{k(\mathbf{x} - \mathbf{x}_i;\ \boldsymbol{\beta})}{\sum_j k(\mathbf{x} - \mathbf{x}_j;\ \boldsymbol{\beta})}. \tag{2.1}$$

Here, $k(\mathbf{x};\ \boldsymbol{\beta})$ is a multivariate version of a density kernel function $k(x;\ \beta)$ with a *bandwidth* parameter $\beta$ and the properties:

$$\int k(x;\ \beta)dx = 1 \qquad \text{and} \qquad \int xk(x;\ \beta)dx = 0 \tag{2.2}$$

The choice of the kernel $k(\cdot)$ and the corresponding bandwidths $\boldsymbol{\beta} = \operatorname{diag}(\beta_1, \ldots, \beta_q)$ are the only parameters of the estimator. However, the kernel shape is of relatively low

importance (Rosenblatt, 1971; Härdle & Marron, 1985) and thus, only the bandwidths remain as crucial parameters.

Popular univariate density kernels are e.g. the Gaussian kernel:

$$k(x;\ \beta) = \exp\left(-\frac{1}{2\beta^2}x^2\right), \tag{2.3}$$

the Epanechnikov kernel ($\beta = 1$), which is continuous, but not differentiable in $|x| = 1$:

$$k(x) = \frac{3}{4}\left[1 - x^2\right]_+ = \begin{cases} \frac{3}{4}\left(1 - x^2\right) & |x| < 1 \\ 0 & |x| \geq 1, \end{cases} \tag{2.4}$$

and the Quartic kernel ($\beta = 1$) which (up to a constant scaling) corresponds to the square of the Epanechnikov kernel and is differentiable in any point:

$$k(x) = \frac{15}{16}\left[1 - x^2\right]_+^2 = \begin{cases} \frac{15}{16}\left(1 - x^2\right)^2 & |x| < 1 \\ 0 & |x| \geq 1. \end{cases} \tag{2.5}$$

The corresponding multivariate versions of the described density kernels can be obtained e.g. by using the product of the univariate kernels applied to each dimension separately:

$$k(\mathbf{x}) \propto \prod_{i=1}^{q} k(x_i) \tag{2.6}$$

Note, that for the evaluation of the Nadaraya-Watson estimator, pre-factors of the kernels can be omitted since any factor cancels out in the ratio of Equation (2.1). Furthermore, the Epanechnikov and the Quartic kernels have finite support which is computational advantageous for the evaluation of the estimator function.

### 2.1.2. Derivation of the UKR model

In order to derive an unsupervised formulation of the Nadaraya-Watson estimator, the samples of the two random variables are seen from a manifold learning viewpoint as *latent* representations[1] $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) \in \mathbb{R}^{q \times N}$ of the corresponding observations $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N) \in \mathbb{R}^{d \times N}$ whereas usually $q < d$. According to the GRF, these latent representations $\mathbf{X}$ are treated as input data which take the role of *parameters* to the regression function (2.1). The resulting estimator then, in correspondence with the Nadaraya-Watson estimator, constitutes the mapping $\mathbf{f} : \mathcal{X} \to \mathcal{Y}$ from the low-dimensional latent space into the higher-dimensional observation space[2]:

$$\mathbf{f}(\mathbf{x}; \mathbf{X}) = \sum_{i=1}^{N} \mathbf{y}_i \frac{k(\mathbf{x} - \mathbf{x}_i)}{\sum_j k(\mathbf{x} - \mathbf{x}_j)}. \tag{2.7}$$

[1] In the following, for the convenience of the reader, the notations $\mathbf{X}$ and $\mathbf{Y}$ will be overloaded and used both for the latent/observation data in matrix form, i.e. $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) \in \mathbb{R}^{q \times N}$ and $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N) \in \mathbb{R}^{d \times N}$ as well as for the same data in the form of sets, i.e. $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ and $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$. The disambiguation of the two forms automatically results from the specific use cases.

[2] In the following, the notation $\hat{\mathbf{y}} = \mathbf{f}(\mathbf{x})$ will be used for the mapping of a single latent parameter $\mathbf{x}$, and the notation $\hat{\mathbf{Y}} = \mathbf{f}(\mathbf{X})$ where $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ will be used to describe the mappings of all latent parameters $\mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)$ of the set $\mathbf{Y}$ *individually*, resulting in a set of the corresponding mappings $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_N\}$.

Since the scaling and the positioning of the parameters $\mathbf{X}$ is generally free, the formerly crucial bandwidth parameters $\boldsymbol{\beta}$ can be set to unit bandwidth (every different bandwidth can be accounted for by an adequate scaling: $\mathbf{X} \to \boldsymbol{\beta}^{-1}\mathbf{X}$) and therefore has been omitted in the formulation above.

The ratio in Equation (2.7) can be conveniently written as basis function $b_i(\mathbf{x}; \mathbf{X})$ centred at $\mathbf{x}_i$ and evaluated at $\mathbf{x}$:

$$b_i(\mathbf{x}; \mathbf{X}) = \frac{k(\mathbf{x} - \mathbf{x}_i)}{\sum_j k(\mathbf{x} - \mathbf{x}_j)}, \tag{2.8}$$

By comprising all of such basis functions in a vector

$$\mathbf{b}(\mathbf{x}; \mathbf{X}) = (b_1(\cdot),\ b_2(\cdot),\ \ \ldots, b_N(\cdot))^T \in \mathbb{R}^N, \tag{2.9}$$

the estimator (2.7) can be denoted in a familiar form:

$$\mathbf{f}(\mathbf{x}; \mathbf{X}) = \sum_{i=1}^{N} \mathbf{y}_i b_i(\mathbf{x}; \mathbf{X}) = \mathbf{Y}\mathbf{b}(\mathbf{x}; \mathbf{X}). \tag{2.10}$$

From this formulation, a special characteristics of the UKR approach can be seen very nicely: Whereas many approaches to supervised and unsupervised learning rely on functions of the same form "matrix $\times$ vector of basis functions" with the matrix containing the weights for linear combination of the basis functions (see (Meinicke, 2000) for an overview), in contrast to most of the other learning approaches, UKR uses *fixed* weights, namely the observations $\mathbf{Y}$, and realises its flexibility to adapt the manifold representation to the underlying data by adapting the basis functions $\mathbf{b}(\mathbf{x}; \mathbf{X})$, i.e. optimising the positioning of the basis function centres $\mathbf{X}$.

In other words, the learning of an UKR manifold does not consist of adapting a matrix of weights such that a fixed set of basis functions nicely represents the underlying observations, but of adapting the basis functions such that the fixed weighting realises a linear combination of them which yields a good estimator for the output data.

Using Equation (2.10) as mapping from the $q$-dimensional latent space into the $d$-dimensional observation data space of the model (and assuming a preselected kernel), a UKR model is fully defined by the observations $\mathbf{Y}$ (which are to be represented and thus given beforehand) and their latent representations $\mathbf{X}$. For a faithful representation of the observations that also generalises to new samples of the underlying distribution, however, $\mathbf{X}$ has to be chosen or rather optimised in an adequate way. Before detailing an automatic optimisation scheme in Section 2.1.4, the next section introduces the term of the *UKR manifold*.

### 2.1.3. The UKR Manifold

The formerly described UKR model provides a (usually) low-dimensional set of latent parameters or *positions*, a set of (usually) higher-dimensional observations and a mapping from latent to observation space. Taking these three ingredients, one can think of a low-dimensional structure or *manifold* that can be described by low-dimensional positions within the structure and which is embedded in a higher-dimensional space. By means of the mapping, the internal position within the structure can be *translated* to its global position in the embedding space.

Thus, a UKR model consisting of data $\mathbf{X} \in \mathbb{R}^{q \times N}$ and $\mathbf{Y} \in \mathbb{R}^{d \times N}$ and a mapping $\mathbf{f}\colon\ \mathbb{R}^q \to \mathbb{R}^d$ constitutes a "UKR manifold" which is described by all points in the embedding

space that, at the same time, belong to the embedded structure. Thus, such a manifold $\mathcal{M}$ can be denoted as the range of $\mathbf{f}(\mathbf{x};\mathbf{X})$, whereas its latent domain needs to be restricted to an appropriate finite subset $\mathcal{D}_x \subset \mathbb{R}^q$:

$$\mathcal{M} = \{\mathbf{y} = \mathbf{f}(\mathbf{x};\mathbf{X}) \mid \mathbf{x} \in \mathcal{D}_x\}. \tag{2.11}$$

Moreover, this means that the manifold unfolds in the span of the "training samples" $\mathbf{Y}$, since for any $\mathbf{x} \in \mathcal{D}_x$, the sum over the weightings of all $\mathbf{y}_i$ in Equation (2.10) is 1 – i.e. $\sum_i b_i(\mathbf{x};\mathbf{X}) = 1$ – and thus yields always a convex combination of the underlying $\{\mathbf{y}_i\}$. The shape and smoothness of this unfolding however heavily depends on the $\mathbf{X}$ and it is not guaranteed that it realises a good generalisation of the observations.

### 2.1.4. UKR optimisation approach

**Reconstruction error.** A straightforward approach to optimising the latent parameters $\mathbf{X}$ of a UKR model is borrowed from the supervised learning scheme and consists of minimising the mean squared reconstruction error:

$$R(\mathbf{X}) = \frac{1}{N}\sum_i \parallel \mathbf{y}_i - \mathbf{f}(\mathbf{x}_i;\mathbf{X}) \parallel^2 = \frac{1}{N} \parallel \mathbf{Y} - \mathbf{Y}\mathbf{B}(\mathbf{X}) \parallel_F^2 \tag{2.12}$$

where

$$\mathbf{B}(\mathbf{X}) = (\mathbf{b}(\mathbf{x}_1;\mathbf{X}),\ \mathbf{b}(\mathbf{x}_2;\mathbf{X}),\ \ldots\ ,\ \mathbf{b}(\mathbf{x}_N;\mathbf{X})) \tag{2.13}$$

is the matrix containing the basis function vectors (2.9) as columns. As denoted in Equation (2.12), $R(\cdot)$ only depends on the choice of $\mathbf{X}$ since the observations $\mathbf{Y}$ are fixed.

In this form, $R(\mathbf{X})$ provides a measure for the quality of the evaluated $\mathbf{X}$ with respect to the reconstruction of the observations $\mathbf{Y}$, which play the role of the training data. The ability of the corresponding UKR manifold to generalise to new data, however, cannot be captured by this formulation.

**Trivial minimisation solution and regularisation.** Indeed, minimising $R(\mathbf{X})$ with respect to $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$ without any regularisation yields a trivial solution for $\mathbf{X}$ with $R(\mathbf{X}) = 0$. Since the $\mathbf{x}_i$ represent centres of kernels $k(\cdot)$ and these kernels are density functions whose outputs vanish for increasing inputs, $R(\cdot)$ is trivially minimised by moving the $\mathbf{x}_i, i = 1 \ldots N$ infinitely apart from each other, i.e. $\forall (i \neq j): \quad \parallel \mathbf{x}_i - \mathbf{x}_j \parallel \ \rightarrow \infty$. In this case, $k(\mathbf{x}_i - \mathbf{x}_j) \rightarrow \delta_{ij} k(\mathbf{0})$ and consequently $b_i(\mathbf{x}_j;\mathbf{X}) \rightarrow \delta_{ij}$. With Equation (2.10), this yields for each reconstruction of $\mathbf{y}_i$ a perfect result $\mathbf{f}(\mathbf{x}_i;\mathbf{X}) = \mathbf{y}_i$ and the overall reconstruction error $R(\cdot)$ becomes 0.

In order to avoid the trivial solution, Klanke (2007) proposes several regularisation approaches which include constraining the extension of the latent domain $\mathcal{D}_x$ or incorporating a lower bound for the latent space density.

**The leave-one-out cross-validation regularisation.** The most remarkable means of UKR for regularisation is to include leave-one-out cross-validation (LOO-CV) in the optimisation process in a very elegant way. With LOO-CV, the reconstructions $\mathbf{f}(\cdot)$ of the observations $\mathbf{y}_i$ in Equation (2.12) are computed without including the corresponding observation $\mathbf{y}_i$ itself. This automatically requires the remaining $\mathbf{x}_j$ to "stay together" in order to yield a

faithful reconstruction of $\mathbf{y}_i$. The modified version of the reconstruction error $R(\cdot)$ can be denoted as:

$$R_{CV}(\mathbf{X}) = \frac{1}{N}\sum_i \parallel \mathbf{y}_i - \mathbf{f}_{-i}(\mathbf{x}_i;\mathbf{X}) \parallel^2 = \frac{1}{N} \parallel \mathbf{Y} - \mathbf{Y}\mathbf{B}_{CV}(\mathbf{X}) \parallel_F^2 \tag{2.14}$$

where $\mathbf{f}_{-i}(\mathbf{x};\mathbf{X})$ is the mapping (2.7) without inclusion of $\mathbf{y}_i$ / $\mathbf{x}_i$, respectively:

$$\mathbf{f}_{-i}(\mathbf{x};\mathbf{X}) = \sum_{l\neq i} \mathbf{y}_l \frac{k(\mathbf{x}-\mathbf{x}_l)}{\sum_{j\neq i} k(\mathbf{x}-\mathbf{x}_j)}. \tag{2.15}$$

$\mathbf{B}_{CV}(\mathbf{X})$ can be easily obtained from $\mathbf{B}(\mathbf{X})$ in Equation (2.13) by zeroing the diagonal elements $b_i(\mathbf{x}_i;\mathbf{X})$ before normalising the column sums to 1.

Most remarkably, thus, whereas LOO-CV usually implies extensive computational costs, the inclusion in the UKR optimisation is computationally free.

**Gradient descent optimisation** For optimising a UKR model by minimising the corresponding objective function – e.g. $R_{CV}(\mathbf{X})$ – with respect to $\mathbf{X}$, Klanke (2007) proposes to use gradient descent optimisation applying the very fast RPROP algorithm (Riedmiller & Braun, 1993; Igel & Hüsken, 2000). This method provides one step size for each parameter of the objective function and adapts them according to the temporal behaviour of the partial derivatives: if a derivative changes the sign in two successive steps, its corresponding step size $\delta$ is decreased by a small amount $\delta^-$, and increased by a small $\delta^+$ otherwise if there is no change of sign. Here, $\delta^-$ and $\delta^+$ are fixed scalars and thus independent of the actual value of the derivative.

However, since $R(\mathbf{X})$ (2.12) as well as $R_{CV}(\mathbf{X})$ (2.14) are highly non-convex functions, an optimisation with gradient-based methods is likely to get stuck in a poor local minimum. To avoid this undesirable case, UKR largely benefits from an adequate initialisation of the latent parameters.

Whereas finding a good initialisation for (manifold) learning approaches often poses an intricate problem in its own right, UKR can easily incorporate the result of a nonlinear spectral embedding method like Isomap (Tenenbaum et al., 2000) or LLE (Roweis & Saul, 2000).

### 2.1.5. Initialisation with spectral embedding methods

Nonlinear spectral embedding methods are powerful tools to uncover intrinsic structures of datasets by providing lower-dimensional coordinates $\{\mathbf{x}_i\}$ of a dataset $\{\mathbf{y}_i\}$ of $N$ observations. In particular, the identification of these coordinates is performed on the basis of a matrix eigendecomposition (or *spectral* decomposition) which do not suffer from local minima and yield a – up to scaling, rotation and translation – unique solution for the embedding. The matrix is of size $N \times N$ and describes, depending on the specific method, a particular point-to-point characteristics or feature of the neighbourhoods of the $\mathbf{y}_i$'s.

Consequently, as a first step, any of these methods builds up a neighbourhood-graph of the dataset which holds the $\{\mathbf{y}_i\}$ as vertices. Edges are placed such that each observation is connected to its $K$ nearest neighbours in the dataset (or, alternatively, to all neighbours within a radius $\varepsilon$ around it).

**Locally Linear Embedding (LLE)** The Locally Linear Embedding Algorithm (Roweis & Saul, 2000) implicitly assumes a manifold-like structure in the underlying dataset and follows the idea that each of the observed data samples $\mathbf{Y} = \{\mathbf{y}_i\}$ together with a small set of its neighbours can describe a small locally linear patch of that manifold. Consequently, each $\mathbf{y}_i$ is expressed as a linear combination $\sum_j w_{ij}\mathbf{y}_j$ of its neighbours. The coefficient matrix $\mathbf{W} = (w_{ij})$ is optimised by minimising the reconstruction error

$$R_{\mathbf{Y}}(\mathbf{W}) = \sum_{i=1}^{N} \left\| \mathbf{y}_i - \sum_{j=1}^{N} w_{ij}\mathbf{y}_j \right\|^2 \tag{2.16}$$

subject to: (1) $w_{ij} = 0$, if $\mathbf{y}_i$ and $\mathbf{y}_j$ are not neighbours (i.e. are not directly connected in the aforementioned neighbourhood-graph) and (2) $\sum_j w_{ij} = 1$, i.e. the reconstruction is a convex combination of the neighbours. The optimised weights then reflect the intrinsic geometric properties of each neighbourhood which are invariant to rotations, translations and scaling.

LLE now aims at transferring these intrinsic properties of the manifold encoded in $\mathbf{W}$ from the high-dimensional representation in form of the observations $\{\mathbf{y}_i\}$ to a lower-dimensional representation in form of corresponding manifold coordinates $\mathbf{X} = \{\mathbf{x}_i\}$. To this end, each $\mathbf{x}_i$ is again expressed as linear combination $\sum_j w_{ij}\mathbf{x}_j$ of its neighbours. However, this time, the $w_{ij}$ are given as the weights obtained from the previous optimisation (reflecting the intrinsic properties of the high-dimensional observations). Thus, the corresponding reconstruction error

$$R_{\mathbf{X}}(\mathbf{X}) = \sum_{i=1}^{N} \left\| \mathbf{x}_i - \sum_{j=1}^{N} w_{ij}\mathbf{x}_j \right\|^2 \tag{2.17}$$

is minimised with respect to the coordinates $\mathbf{X}$ in order to obtain a faithful lower-dimensional realisation of the neighbourhood properties encoded in $\mathbf{W}$. The resulting $\mathbf{X}$ afterwards holds these optimised low-dimensional coordinates.

**Isomap** The Isomap algorithm (Tenenbaum et al., 2000) defines the underlying $N \times N$ matrix as distance matrix $\mathbf{D}$ holding the distances $D_{ij}$ between two observations $\mathbf{y}_i$ and $\mathbf{y}_j$ in the neighbourhood-graph. These "geodesic" distances are determined as the lengths of the shortest paths in the graph (if no path exists, $D_{ij}$ is undefined) and can be efficiently calculated e.g. by means of the classical algorithm presented by Dijkstra (1959). The subsequent embedding of the lower-dimensional coordinates $\{\mathbf{x}_i\}$ then is performed with classical Multi-Dimensional Scaling (MDS; e.g. (Cox & Cox, 2000)).
Due to its foundation on the combination of two well-studied basic principles, namely the Dijkstra algorithm and the MDS, Isomap is a very transparent approach which, at the same time, provides computationally very efficient solutions. In addition, since a fast implementation of Isomap for MATLAB[3] is available on the web[4] and the author experienced very convincing results in combination with the data processed in this thesis, Isomap has been the spectral embedding method of choice for all evaluations presented in the remainder of this thesis.

[3] `http://www.mathworks.com/`
[4] `http://waldron.stanford.edu/~isomap/`

As described above, spectral embedding methods are very powerful tools when it comes to embedding data into a low-dimensional space and to uncovering intrinsic manifold-like structures in datasets. It is however important to notice that, even if these methods implicitly base on the assumption of an existing manifold structure, they do not provide an explicit manifold model, neither a mapping between manifold space and observation space. They "only" provide faithful low-dimensional coordinates of the given "training data". Furthermore, depending on the characteristics of the observation dataset, the quality of the resulting embeddings can heavily depend on the choice of the neighbourhood parameter $K$ (or $\varepsilon$) of the neighbourhood-graph: if chosen too small, the graph may be split into several isolated subgraphs, if chosen too large, so called "short-cut connections" in the graph may appear which connect data points that are not neighbours in the underlying true manifold structure. Both effects negatively influence the embedding outcome up to complete uselessness, and whereas the first case can easily be encountered from the resulting graph(s), the latter may be hard to identify in a real data scenario.

Nevertheless, the embedding characteristics yield a perfect combination with UKR which, on the one hand, can largely benefit from an initialisation of the latent variables resulting from spectral embedding methods and, on the other hand, provides an explicit manifold model including mapping from latent to observation space which is missing in the spectral embedding approach.

**UKR initialisation with spectral embeddings** As described in Section 2.1.2, the initialisation of a UKR model only requires to specify an initial set of latent parameters $\mathbf{X}_{init}$. Therefore, the outcome of a spectral embedding, for instance $\mathbf{X}_{spec}$, can be easily included in the UKR optimisation by finding an adequate scaling of the $\mathbf{X}_{spec}$ that yields the smallest error for the chosen UKR reconstruction error $R^\star(\cdot)$, e.g. $R(\cdot)$ (2.12) or $R_{CV}(\cdot)$ (2.14):

$$\mathbf{X}_{init} = \mathrm{diag}(\mathbf{s}_{opt})\mathbf{X}_{spec} \quad \text{with} \quad \mathbf{s}_{opt} = \arg\min_{\mathbf{s}} R^\star\left( \ \mathrm{diag}(\mathbf{s})\mathbf{X}_{spec} \ \right), \tag{2.18}$$

where $\mathbf{s}$ or $\mathbf{s}_{opt}$ are the $q$-dimensional scale factors.

Moreover, Klanke (2007) states that this method can also be used to evaluate the appropriateness of the underlying neighbourhood parameter of the spectral embedding. Performing the scaling for a set of candidate parameters resulting in $\{\mathbf{X}^i_{spec}\}$ and choosing the neighbourhood parameter which yields the smallest reconstruction error $R^\star(\cdot)$ thus can be used to automatically parametrise the spectral embedding.

### 2.1.6. Projection from observation space to UKR latent space

Whereas the mapping $\mathbf{f} : \mathcal{D}_x \to \mathcal{M}$ (2.7) from the latent domain $\mathcal{D}_x$ into the manifold subspace $\mathcal{M} \subset \mathbb{R}^d$ in observation space is inherently part of the UKR model, a mapping $\mathbf{g} : \mathbb{R}^d \to \mathcal{D}_x$ that yields the latent position on the manifold corresponding to a data point in observation space is not directly supported. Instead, an orthogonal projection onto the UKR manifold can be used as pendant[5]:

$$\mathbf{x}^\star = \mathbf{g}(\mathbf{y}; \mathbf{X}) = \arg\min_{\mathbf{x} \in \mathcal{D}_x} \| \ \mathbf{y} - \mathbf{f}(\mathbf{x}; \mathbf{X}) \ \|^2 . \tag{2.19}$$

[5] In the following, the notation $\mathbf{x}^\star = \mathbf{g}(\mathbf{y})$ will be used for the projection of a single observation $\mathbf{y}$, and the notation $\mathbf{X}^\star = \mathbf{g}(\mathbf{Y})$ where $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$ will be used to describe the projections of all observations $\mathbf{x}^\star_i = \mathbf{g}(\mathbf{y}_i)$ of the set $\mathbf{Y}$ *individually*, resulting in a set of corresponding latent parameters $\mathbf{X}^\star = \{\mathbf{x}^\star_1, \ldots, \mathbf{x}^\star_N\}$

### 2.1.7. Overview of the whole UKR optimisation scheme

Summarising the previous sections, the definition and optimisation of a UKR model (in its basic form) in order to find an appropriate lower-dimensional latent representation $\mathbf{X}$ of a set of observations $\mathbf{Y}$ includes the following steps:

1. Compute a set $\{\mathbf{X}^i_{spec}\}$ of spectral embeddings corresponding to a set of candidate parameters of the underlying neighbourhood graph (see Section 2.1.5).

2. Find the optimal scalings $\mathbf{s}^i_{opt}$ for each $\mathbf{X}^i_{spec}$ such that the chosen UKR reconstruction error $R^\star(\mathrm{diag}(\mathbf{s}^i_{opt})\mathbf{X}^i_{spec})$ is minimal (see Section 2.1.5) and build up a set of candidate UKR initialisations according to the optimal scalings:

$$\mathbf{X}^i_{init} = \mathrm{diag}(\mathbf{s}^i_{opt})\mathbf{X}^i_{spec}.$$

3. Choose the $\mathbf{X}^k_{init}$ in $\{\mathbf{X}^i_{init}\}$ that yields the minimal UKR reconstruction error $R^\star(\cdot)$ as latent initialisation of the UKR model:

$$\mathbf{X}_{init} = \mathbf{X}^k_{init} \quad \text{with} \quad k = \arg\min_i R^\star(\mathbf{X}^i_{init}).$$

4. Starting with $\mathbf{X} = \mathbf{X}_{init}$ as latent parameters, optimise $\mathbf{X}$ by minimising the chosen UKR reconstruction error with respect to the latent parameters, e.g. with gradient-based optimisation (see Section 2.1.4):

$$\mathbf{X}_{final} = \arg\min_{\mathbf{X}} R^\star(\mathbf{X}).$$

The optimised UKR model then consists of the given observations $\mathbf{Y}$, the corresponding optimised latent parameters $\mathbf{X}_{final}$ and the parametrised mapping $\mathbf{y} = \mathbf{f}(\mathbf{x}; \mathbf{X}_{final})$ to calculate the position $\mathbf{y}$ in the observation space of the evaluated latent coordinate $\mathbf{x}$ within the manifold.

## 2.2. Gaussian Mixture Model and Gaussian Mixture Regression

Recently, the combination of Gaussian Mixture Model (GMM) and Gaussian Mixture Regression (GMR) has been used in several approaches to motion trajectory generalisation in the context of imitation learning in robotics (Calinon et al., 2006, 2007; Chernova & Veloso, 2007; Mühlig, Gienger, Steil, & Goerick, 2009; Mühlig, Gienger, Hellbach, et al., 2009).

In this combination, the GMM takes the part of modelling the observed motion trajectories. The GMR is used to extract a generalised form of the underlying training trajectories that can be used e.g. for the actuation of the motion on a robotic system. As a special feature of the GMR, the method inherently provides a time-dependent measure for the confidence of the generated trajectory.

### 2.2.1. Gaussian Mixture Model (GMM)

A *Gaussian Mixture Model* (GMM) is a probabilistic representation of a set of observed input/output samples $\{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$ where $\mathbf{z}_i = (\mathbf{x}_i^T, \mathbf{y}_i^T)^T$ with inputs $\mathbf{x}_i \in \mathbb{R}^q$ and outputs $\mathbf{y}_i \in \mathbb{R}^d$. It is an example of a finite mixture model (e.g. (McLachlan & Peel, 2000)) and is based on a finite mixture density.

A finite mixture density $p(\mathbf{z})$ is a probability density function that is defined as a mixture of $K$ other probability density functions

$$p(\mathbf{z}) = \sum_{k=1}^{K} \alpha_k p(\mathbf{z}|k) \tag{2.20}$$

where $\alpha_k \geq 0$ with $\sum_{k=1}^{K} \alpha_k = 1$ are the mixing coefficients and $p(\mathbf{z}|k)$ are $(q+d)$-dimensional conditional probability distributions.

In the case of the GMM, these conditional probabilities correspond to Gaussian distributions $g_k$ with parameters $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ whereas $\boldsymbol{\mu}_k$ is the mean vector and $\boldsymbol{\Sigma}_k$ the covariance matrix:

$$p(\mathbf{z}) = \sum_{k=1}^{K} \alpha_k g_k(\mathbf{z}) \tag{2.21}$$

$$g_k(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{2.22}$$

$$= \frac{1}{\sqrt{(2\pi)^{(q+d)}|\Sigma_k|}} \exp\left(-\frac{1}{2}(\mathbf{z}-\boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\mathbf{z}-\boldsymbol{\mu}_k)\right). \tag{2.23}$$

A GMM with $K$ components is therefore defined by the set of mixing coefficients and the parameters to the Gaussian distributions, i.e. $\{\alpha_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$.

The fitting of the model to a set of training data can be performed by maximising the log-likelihood

$$L = \sum_{i=1}^{N} \log p(\mathbf{z}_i) = \sum_{i=1}^{N} \log \sum_{k=1}^{K} \alpha_k p(\mathbf{z}|k). \tag{2.24}$$

with respect to the model parameters. This can be efficiently done with the *Expectation-Maximisation* (EM) algorithm (Dempster et al., 1977) which iteratively maximises the log-likelihood by performing two alternating steps for all components $k = 1 \ldots K$:

1. *Expectation.* Calculate the a-posteriori probabilities ('Expectations') of the training data points $\mathbf{z}_i$ according to the model components $k$:

$$P(k|\mathbf{z}_i) = \frac{\alpha_k p(\mathbf{z}_i|k)}{\sum_{l=1}^{K} \alpha_l p(\mathbf{z}_i|l)} = \frac{\alpha_k g_k(\mathbf{z}_i)}{\sum_{l=1}^{K} \alpha_l g_l(\mathbf{z}_i)} \tag{2.25}$$

2. *Maximisation.* Optimise the model parameters $\{\alpha_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$ by maximising the log-likelihood of the model considering the a-posteriori probabilities from the expec-

tation step. The maximisation yields for the new parameters $\alpha'_k, \boldsymbol{\mu}'_k, \boldsymbol{\Sigma}'_k$:

$$\alpha'_k = \frac{1}{N}\sum_{i=1}^{N} P(k|\mathbf{z}_i) \tag{2.26}$$

$$\boldsymbol{\mu}'_k = \frac{\sum_{i=1}^{N} P(k|\mathbf{z}_i)\mathbf{z}_i}{\sum_{i=1}^{N} P(k|\mathbf{z}_i)} \tag{2.27}$$

$$\boldsymbol{\Sigma}'_k = \frac{\sum_{i=1}^{N} P(k|\mathbf{z}_i)(\mathbf{z}_i - \boldsymbol{\mu}_k)(\mathbf{z}_i - \boldsymbol{\mu}_k)^T}{\sum_{i=1}^{N} P(k|\mathbf{z}_i)}. \tag{2.28}$$

The monotone increase of the model's likelihood with respect to the training set is guaranteed for each EM step. In order to initialise the model parameters, the *k-means* algorithm can be performed and initial model parameters can be derived from the resulting clusters.

Since the algorithm performs a local optimisation, it can however get stuck in local maxima of the likelihood function. In addition, the optimal number of mixture components $K$ (and thus also the number of initial clusters, respectively) cannot be directly derived from the training data. One approach is to estimate a set of models for an increasing number of mixture components and to select the best model according to some optimality criterion. Vlassis and Likas (2002) propose e.g. the use of cross-validation with a set of test points, Calinon et al. (2007) and Mühlig, Gienger, Steil, and Goerick (2009) rely on the *Bayesian Information Criterion* (Schwarz, 1978).

### 2.2.2. Gaussian Mixture Regression (GMR)

After having modelled a set of training data $\{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$ with a GMM as described in the previous section, the *Gaussian Mixture Regression* (GMR) can be used to reconstruct a generalised form of the underlying distribution.

In the application of the GMM to modelling trajectories of consecutive signals $\{\mathbf{z}^s_1, \ldots, \mathbf{z}^s_N\}$ (e.g. joint angles of a robot and end-effector positions during a motion), the training data are usually associated with a temporal value $z^\tau_t$ corresponding to the temporal position $t$ within this set. Together with the original signal part $\mathbf{z}^s_t$, the temporal information $z^\tau_t$ makes part of the actual training data:

$$\mathbf{z}'_t = \begin{bmatrix} z^\tau_t \\ \mathbf{z}^s_t \end{bmatrix} \text{ with } z^\tau_1 < z^\tau_2 < \cdots < z^\tau_N \tag{2.29}$$

The trained GMM therefore also contains temporal information about the modelled data which can be used to reconstruct a generalised trajectory.

To this end, the parameters $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ of a given GMM $\{\alpha_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$ with $K$ mixture components are written corresponding to the temporal parts and the signal parts, respectively:

$$\boldsymbol{\mu}_k = \begin{bmatrix} \mu^\tau_k \\ \boldsymbol{\mu}^s_k \end{bmatrix} \quad , \quad \boldsymbol{\Sigma}_k = \begin{bmatrix} \Sigma^\tau_k & \boldsymbol{\Sigma}^{\tau s}_k \\ \boldsymbol{\Sigma}^{s\tau}_k & \boldsymbol{\Sigma}^s_k \end{bmatrix} \tag{2.30}$$

For a given temporal query point ("time step") $z^{\tau\star}$, the GMM can be evaluated to compute the corresponding model responses for the signal part $\hat{\mathbf{z}}^s$ by mixture regression.

To this end, the conditional expectation $\hat{\mathbf{z}}_k^s$ for query point $z^{\tau\star}$ and the corresponding estimated covariance matrix $\hat{\mathbf{\Sigma}}_k^s$ is computed for every mixture component $k$:

$$\hat{\mathbf{z}}_k^s = \boldsymbol{\mu}_k^s + \mathbf{\Sigma}_k^{s\tau}(\Sigma_k^\tau)^{-1}(z^{\tau\star} - \mu_k^\tau) \tag{2.31}$$

$$\hat{\mathbf{\Sigma}}_k^s = \mathbf{\Sigma}_k^s - \mathbf{\Sigma}_k^{s\tau}(\Sigma_k^\tau)^{-1}(\mathbf{\Sigma}_k^{\tau s}) \tag{2.32}$$

The contribution of each component $k$ is derived from the probability that it is responsible for the modelling of query point $z^{\tau\star}$:

$$\beta_k = \frac{g_k^\tau(z^{\tau\star})}{\sum_{l=1}^K g_l^\tau(z^{\tau\star})}, \tag{2.33}$$

where $g_k^\tau(\cdot)$ corresponds to the Gaussian distribution with mean $\mu_k^\tau$ and variance $\Sigma_k^\tau$. The overall mixtures therefore have the form:

$$\hat{\mathbf{z}}^s = \sum_{k=1}^K \beta_k \hat{\mathbf{z}}_k^s \tag{2.34}$$

$$\hat{\mathbf{\Sigma}}^s = \sum_{k=1}^K \beta_k^2 \hat{\mathbf{\Sigma}}_k^s \tag{2.35}$$

A generalised version of the training data trajectory can therefore be obtained by performing the described regression method for a series of consecutive temporal query points $\{z_t^{\tau\star}\}_{t=t_0}^T$. The resulting trajectory of expected signals $\{\hat{\mathbf{z}}_t^s\}_{t=t_0}^T$ can be used as target trajectory in a reproduction application. The corresponding covariance estimates $\hat{\mathbf{\Sigma}}_t^s$ (or rather the inverse of it) can serve as confidence measure of the expected signal $\hat{\mathbf{z}}_t^s$.

## 2.3. Gaussian Process Latent Variable Model & Gaussian Process Regression

Introduced for the purpose of visualisation by Lawrence (2004, 2005), the Gaussian Process Latent Variable Model (GPLVM) has recently been used in the field of computer graphics in a variety of applications, especially in the context of avatar animation and pose and motion modelling (Grochow et al., 2004; Hou et al., 2007; Bitzer et al., 2008; Ek, Rihan, et al., 2008; Ek, Torr, & Lawrence, 2008). In addition, several modifications to the original method have been proposed (Lawrence & Quiñonero Candela, 2006; Lawrence & Moore, 2007; Urtasun et al., 2008; Bitzer & Vijayakumar, 2009).

### 2.3.1. Gaussian Process Latent Variable Model (GPLVM)

The *Gaussian Process Latent Variable Model* has been introduced by Lawrence (2004) as a non-linear generalisation of probabilistic Principal Component Analysis (PCA). It realises a model of a high-dimensional data set $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N]^T \in \mathbb{R}^{N\times d}$ of observations with corresponding latent parameters $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^T \in \mathbb{R}^{N\times q}$ and a Gaussian Process (GP) mapping from latent to data space. Similar to the GMM/GMR approach in the previous section, this GP "mapping" yields a probability distribution in data space for a given latent

parameter. The "best guess" for the resulting data point therefore constitutes the mean of this distribution along with a measure for its certainty given by the conditional covariance.

The model with parameters $\boldsymbol{\beta}$ (described in the following) provides one GP for each data dimension. The likelihood of $\mathbf{Y}$ given $\mathbf{X}$ can therefore be denoted as:

$$p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\beta}) = \prod_{i=1}^{d} p(\mathbf{y}_{:,i}|\mathbf{X}, \boldsymbol{\beta}), \tag{2.36}$$

where $\mathbf{y}_{:,i}$ represents the $i$-th column of $\mathbf{Y}$ and therefore the vector holding the $i$-th components of all observations $\{\mathbf{y_1}, \ldots, \mathbf{y_N}\}$. The corresponding likelihood is given by

$$p(\mathbf{y}_{:,i}|\mathbf{X}, \boldsymbol{\beta}) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}) \tag{2.37}$$

where $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ denotes a Gaussian distribution with zero mean and covariance matrix $\boldsymbol{\Sigma}$. This covariance matrix is defined by a positive semidefinite covariance function $k(\mathbf{x}_i, \mathbf{x}_j)$:

$$(\boldsymbol{\Sigma})_{ij} = k(\mathbf{x}_i, \mathbf{x}_j). \tag{2.38}$$

The choice of $k(\cdot)$ is usually the same for all data dimensions and therefore for each GP in one GPLVM.
In the case of the (linear) probabilistic PCA, this covariance function is given by:

$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}' + \beta^{-1}\delta(\mathbf{x}, \mathbf{x}'), \tag{2.39}$$

where $\delta(\cdot, \cdot)$ represents the Kronecker delta.

For the non-linear case, the radial basis function is a common choice:

$$k(\mathbf{x}, \mathbf{x}') = \beta_1 \exp\left(-\frac{\beta_2}{2} \parallel \mathbf{x} - \mathbf{x}' \parallel^2\right) + \beta_3^{-1}\delta(\mathbf{x}, \mathbf{x}'). \tag{2.40}$$

Since the covariance function $k(\cdot, \cdot)$ is required to be positive semidefinite, it is at the same time a valid Mercer kernel and the covariance matrix $\boldsymbol{\Sigma}$ can therefore be seen as kernel matrix.

In order to fit the GPLVM to the set of observations $\mathbf{Y}$, the log-likelihood of Equation (2.36) is maximised with respect to the latent variables $\mathbf{X}$ and the kernel parameters $\boldsymbol{\beta}$. The log-likelihood can be denoted as:

$$L = -\frac{dN}{2}\ln 2\pi - \frac{d}{2}\ln|\boldsymbol{\Sigma}| - \frac{1}{2}\operatorname{tr}(\boldsymbol{\Sigma}^{-1}\mathbf{Y}\mathbf{Y}^T) \tag{2.41}$$

In order to perform a gradient-based optimisation, the gradient of $L$ with respect to the kernel (irrespective of the latent variables $\mathbf{X}$ and the kernel parameters $\boldsymbol{\beta}$) can be denoted as:

$$\frac{\partial L}{\partial \boldsymbol{\Sigma}} = \boldsymbol{\Sigma}^{-1}\mathbf{Y}\mathbf{Y}^T\boldsymbol{\Sigma}^{-1} - d\boldsymbol{\Sigma}^{-1}. \tag{2.42}$$

For a particular kernel, $\frac{\partial L}{\partial \boldsymbol{\Sigma}}$ can then be combined with the kernel gradients $\frac{\partial \boldsymbol{\Sigma}}{\partial x_{ij}}$ and $\frac{\partial \boldsymbol{\Sigma}}{\partial \beta_i}$, respectively, through the chain rule.

As a convenient non-linear solver, Lawrence (2005) suggests to apply the scaled conjugate gradients method introduced by Møller (1993).

### 2.3.2. Gaussian Process Regression (GPR)

If a GPLVM is successfully trained for a set of observations $\mathbf{Y} = [\mathbf{y_1}, \ldots, \mathbf{y}_N]^T \in \mathbb{R}^{N \times d}$ as described in the previous section, the appropriate latent variables $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^T \in \mathbb{R}^{N \times q}$ and parameters $\boldsymbol{\beta}$ of the covariance function ('kernel') $k(\cdot, \cdot)$ can be used for the regression of new data points $\mathbf{Y}^\star = \{\mathbf{y}_1^\star, \ldots, \mathbf{y}_{N^\star}^\star\}$.

To this end, the *Gaussian Process Regression* can be applied for a set of desired latent query points $\mathbf{X}^\star = \{\mathbf{x}_1^\star, \ldots, \mathbf{x}_{N^\star}^\star\}$. Corresponding to the design of the GPLVM, each dimension $j$ of the $\mathbf{y}_i^\star$ is modelled as one independent GP, the corresponding components $y_{i,j}^\star$ are recovered separately.

A GP model is based on the assumption that the underlying observations $\mathbf{y}_{:,i}$ can be represented as a sample from a multivariate Gaussian distribution (where $\mathbf{y}_{:,i}$ again represents the $i$-th column of the training data matrix $\mathbf{Y}$; see previous section):

$$\mathbf{y}_{:,i} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}), \tag{2.43}$$

where $\boldsymbol{\Sigma}$ is the $N \times N$ data kernel matrix defined by $(\boldsymbol{\Sigma})_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$.

For the observation vector extended by new observations $\mathbf{y}_{:,i}^\star$ that are to be estimated through regression, it can therefore be denoted:

$$\begin{bmatrix} \mathbf{y}_{:,i} \\ \\ \mathbf{y}_{:,i}^\star \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}_\star^T \\ \\ \boldsymbol{\Sigma}_\star & \boldsymbol{\Sigma}_{\star\star} \end{bmatrix}\right), \tag{2.44}$$

where $\boldsymbol{\Sigma}$ is again the $N \times N$ data kernel matrix, $\boldsymbol{\Sigma}_\star$ holds the $N^\star \times N$ 'mixed' covariances of training and target data, i.e. $(\boldsymbol{\Sigma}_\star)_{ij} = k(\mathbf{x}_i^\star, \mathbf{x}_j)$ and $\boldsymbol{\Sigma}_{\star\star}$ holds the $N^\star \times N^\star$ pure target data covariances, i.e. $(\boldsymbol{\Sigma}_{\star\star})_{ij} = k(\mathbf{x}_i^\star, \mathbf{x}_j^\star)$.

The conditional probability $p(\mathbf{y}_{:,i}^\star | \mathbf{y}_{:,i})$ can be derived as the Gaussian distribution:

$$\mathbf{y}_{:,i}^\star | \mathbf{y}_{:,i} \sim \mathcal{N}(\boldsymbol{\Sigma}_\star \boldsymbol{\Sigma}^{-1} \mathbf{y}_{:,i} \quad , \quad \boldsymbol{\Sigma}_{\star\star} - \boldsymbol{\Sigma}_\star \boldsymbol{\Sigma}^{-1} \boldsymbol{\Sigma}_\star^T) \tag{2.45}$$

Correspondingly, the best estimate for $\mathbf{y}_{:,i}^\star$ is given by the mean of this distribution:

$$\hat{\mathbf{y}}_{:,i}^\star = \boldsymbol{\Sigma}_\star \boldsymbol{\Sigma}^{-1} \mathbf{y}_{:,i} \tag{2.46}$$

and a measure for the certainty of these values is described by the variances:

$$\boldsymbol{\Sigma}_{:,i}^\star = \boldsymbol{\Sigma}_{\star\star} - \boldsymbol{\Sigma}_\star \boldsymbol{\Sigma}^{-1} \boldsymbol{\Sigma}_\star^T. \tag{2.47}$$

## 2.4. Discussion

From the above described methods, UKR has been identified as the most suitable for the work presented in the remainder of this thesis. The explicit manifold representation of UKR realised on the basis of latent variables of the observed data is perfectly qualified for the targeted representation and control schemes.

On the one hand, the foundation on the unsupervised reformulation of the Nadaraya-Watson kernel estimator provides a rich corpus of analyses since it constitutes one of the standard methods in the field of locally weighted learning (e.g. reviewed by Atkeson et al. (1997a, 1997b)) and has been studied for several decades.

On the other hand, the recent reformulation of the originally supervised approach as unsupervised method introduces interesting new possibilities and aspects of the method and

enables the approach to be investigated from a completely new viewpoint which open up new perspectives and a new field of applications.

The GPLVM has recently become quite popular in the field of avatar animation in the field of computer graphics and has finally also found its way to the domain of robotics. Whereas it can be argued in a similar way for the use of the GPLVM/GPR combination as it has been done for the UKR above, the method however is computationally more demanding ($O(N^3)$ where $N$ is the number of training data vs. $O(dN^2)$ for UKR with $d$ being the dimensionality of the training data).

The combination of GMM and GMR has already been successfully applied to motion representation and also recognition e.g. by Calinon et al. (2007) or Mühlig, Gienger, Steil, and Goerick (2009). The pure GMM approach is computational even more efficient than UKR ($O(KNd^2)$) where $K$ is the number of mixing models which needs to be specified beforehand. While the required number of models can be estimated empirically, the initialisation of the latter however is usually performed with standard *k-means*, which does not necessarily yield meaningful localisations of the models. In addition, the approach does not feature an explicit latent space which renders it intricate to follow the goals of this thesis. The missing flexibility of the method concerning the required structure or form of the data (for training and for the recognition application) also poses problems with the targeted applications. In order to get a faithful representation, a fixed number of observations as well as a fixed sampling rate (i.e. time step size) is required for all training sequences same or needs to be interpolated with auxiliary methods like Dynamic Time Warping (e.g. (Keogh & Pazzani, 2001; Ratanamahatana & Keogh, 2004)). This "time scale correspondence" afterwards constitutes an intrinsic characteristics of the model. Therefore, only observations in the same time scale, i.e. performed in the same speed, and with the same amount of observations as represented can be recognised afterwards. Last but not least, the recognition system cannot be performed with the pure GMM/GMR system, but needs to be combined e.g. with Hidden Markov Models (Calinon et al., 2007), which requires additional training. A *segmentation* of a sequence of multiple motions as it will be presented in Chapter 6 is not possible at all.

All these considerations resulted in the decision for the use of UKR as the method of choice for the main aspects developed in this thesis.

# 3. Hand Posture Acquisition for Anthropomorphic Robotic Hands

The generation and acquisition of sufficiently accurate hand posture data as a basis for learning dextrous robot grasping and manipulation is a complex and time-consuming task. Several analyses of hands and grasps use optical markers to extract hand posture data from camera images (e.g. (Chang et al., 2007; Supuk et al., 2005)) which can provide very accurate joint angle data under optimal conditions. However, these methods often struggle with lighting conditions and work space restrictions. Especially in the case of grasping and object manipulation, occlusions of one or several of the markers pose additional problems. Moreover, vision systems that can provide highly accurate data are very expensive and the application area is often restricted to its installation location.

One alternative to visual hand posture recognition is given by means of data gloves which measure the joint angles directly from sensors implemented in the gloves. Whereas some of the problems that occur using a vision system can be bypassed with this approach, the mapping of raw sensor values to joint angles as well as the calibration of such mapping become more crucial and the choice of the applied methods less obvious. Fischer et al. (1998) use a data glove for the direct control of a four-fingered robot hand. For the glove calibration phase, however, they again used an optical marker solution coupled with a calibrated stereo camera system. The vision system is required to generate a training set of visually extracted fingertip positions together with the corresponding data glove sensor values. A linearly augmented feed-forward neural network is used to learn a representation of the training data in order to realise the mapping from the sensor values to the fingertip positions. Using their methods, single joints as well as the finger segment positions cannot be controlled directly.

Another approach was presented by Chou et al. (2000) who record the finger joint angles extracted from monocular camera pictures along with the corresponding glove data. Subsequently, they performed a least squares regression on the combined data set distinguishing between joints with one-to-one and one-to-two sensor mappings.

Whereas these approaches to glove calibration benefit from the incorporation of the vision systems in the way that they can generate a large amount of highly accurate training data in a relatively short time, they have to struggle with the same problems that the purely vision-based approaches have, at least during the calibration phase, and they also necessitate the need for an adequate vision system.

Other approaches have been presented that do not require additional hardware to the glove itself, but sometimes with some simple auxiliary objects. Griffin et al. (2000) presented an approach to dexterous telemanipulation in which they used a data glove calibration procedure focussed on only the index finger and the thumb. In order to generate training data, the user places the tips of these fingers against each other and then moves them while maintaining the rolling contact. A least squares regression was then carried out for the parameters of a closed kinematic chain model. Turner (2001) extended this approach to four fingers: the thumb and the index, middle and ring finger. In comparison

to the vision-based approaches, Turner reports large position errors for the finger tips and especially problems with calibrating the thumb.

Another vision-free approach was presented by Kahlesz et al. (2004) who follow a different type of calibration goal. The target is a "high visual fidelity" of the sensor-to-angle mapping rather than a highly accurate positioning of the finger tips. They explicitly model cross couplings of abduction and flexion joints by recording isosurfaces in the corresponding flexion/abduction space using simple auxiliary objects. The abduction angles then are computed by a linear mapping of the distances to the isosurfaces.

The method proposed in this chapter constitutes an approach to vision-free data glove calibration. The aim of this work is to allow the use of the data glove for data acquisition in the field of robot grasping and object manipulation. To obtain more accurate data, the raw sensory data from the glove are mapped to joint angles of a physics-based 3D computer model which simulates the grasping or manipulation task. The joint data affected by the collision detection of the physics-based simulation then serve as training data for the learning algorithms.

In this approach, the focus is not on an accurate reproduction of the fingertip positions realised by the data glove (and the hand inside the glove), but on imitating the demonstrated hand posture as a whole. These two approaches may sound similar, however, depending on the robot hand size and kinematic design, they can yield significantly different hand postures and in fact are directed towards two different goals:

The first approach (accurate reproduction of fingertip positions) usually aims at generating data that represent hand postures corresponding to a specific set of contact points on a manipulated object. To this end, the exact fingertip positions need to be recorded whereas the shape and form of the hand posture is neglected.

The second approach (hand posture as a whole) which is followed in this chapter, aims at generating natural looking hand postures and movements (i.e., posture sequences) and neglects the direct task-related correspondence of demonstrated and represented postures in the recorded data. In the following example, turning a bottle cap is used as prototype manipulation movement. Concerning this scenario, the two approaches mentioned above provide training data with distinct characteristics:

1. The first one tries to capture the exact contact points on the bottle cap surface by recording the fingertip positions in space. The resulting data therefore correspond to the task of turning a cap of the same size as demonstrated. The problem to solve is the mapping of the recorded fingertip positions onto the target robot platform. Depending on the difference in size and shape between the demonstrator's hand and the robot's hand, this mapping can produce rather unnatural hand postures or can even render a mapping infeasible, e.g., if the width of the robot fingers is too large and cannot generate two nearby contacts.

2. The second approach tries to imitate the demonstrated hand posture and therefore the relative positioning of the fingers and finger segments. The resulting data can usually not be directly used for the manipulation of the same demonstrated bottle cap, since different hand sizes also change the distance between the fingertips in the manipulation movement. The problem becomes one of finding the cap (radius) that fits the recorded data or, the other way around, to find a set of data that fits the targeted cap size. This can be avoided by performing the manipulation directly with the robot platform and using the data glove only as specialised input device. This procedure is in line with Oztop et al. (2006, 2007, 2008) who propose using the robot


(a) The bend sensors (visible are the fixation stitchings) are placed on top of the single joints. Wrist and thumb roll sensors (red bars) are placed insight the top layer of the glove.

(b) Forefinger / middle finger abduction sensor in zero-abduction position for (nearly) not flexed fore and middle finger.

(c) As in b), the fingers are not abducted, but the abduction sensor is heavily bended/twisted due to the flexion of the middle finger.

**Figure 3.1.:** The Immersion Cyberglove II Wireless. As depicted in (b-c), the bending degree of the finger abduction sensors do not only depend on the finger abduction, but also heavily depend on the flexion of the neighbouring fingers.

hand as tool or even as an *external limb* of the demonstrator who is able to transfer his own dextrous skills to the robot hand in this manner. This idea will be addressed in more detail in the following.

From the mapping or calibration point of view, the approach presented in this chapter is related to the work of Kahlesz et al. (2004). Instead of generating isosurfaces out of data recorded in the calibration phase, the glove calibration presented here is however based on explicit, parameterisable models for the cross coupled sensors of the fingers and the thumb, derived from analyses of recorded sensor data. In this manner, the data glove's intrinsic sensor dependencies caused by the glove design can be captured. By using a flexible data model and fitting it to the recorded data, the user-specific model parameters are reduced to the minimal and maximal sensor values. The calibration itself thus can be performed simply by moving the hand-mounted glove to the extremal joint positions. In addition, since the method attempts to capture the intrinsic design of the glove, values obtained during calibration can be used as a default configuration that can then yield adequate results for many different users and across many different tasks.

As in the previously mentioned works about data gloves (Fischer et al., 1998; Chou et al., 2000; Griffin et al., 2000; Turner, 2001; Kahlesz et al., 2004), the Immersion Cyber-

GLOVE is used – currently the most accurate available data glove. In the Neuroinformatics Group at Bielefeld University, the second version of this is used, which is wireless and has 22 bend sensors including two wrist sensors. These sensors are placed in the top layer of the glove on top of the corresponding finger/hand joints (see Figure 3.1). Each sensor returns integer values in the interval $[0; 255]$, whereas 0 corresponds to no bending and 255 to the maximum bending. Due to the natural joint limits of human finger joints, the whole range of possible values is usually not used.

Before describing the details of the data glove calibration and the mapping from the glove sensory data onto robotic hands, a brief introduction to such hands is given in the next section. This comprises a more detailed description of the two specific robotic hands that are used in this thesis. Subsequently, a simulation model of a human hand is presented that will serve as an intermediate mapping level between the raw sensory data from the data glove and the joint angle representation for the targeted robotic platform.

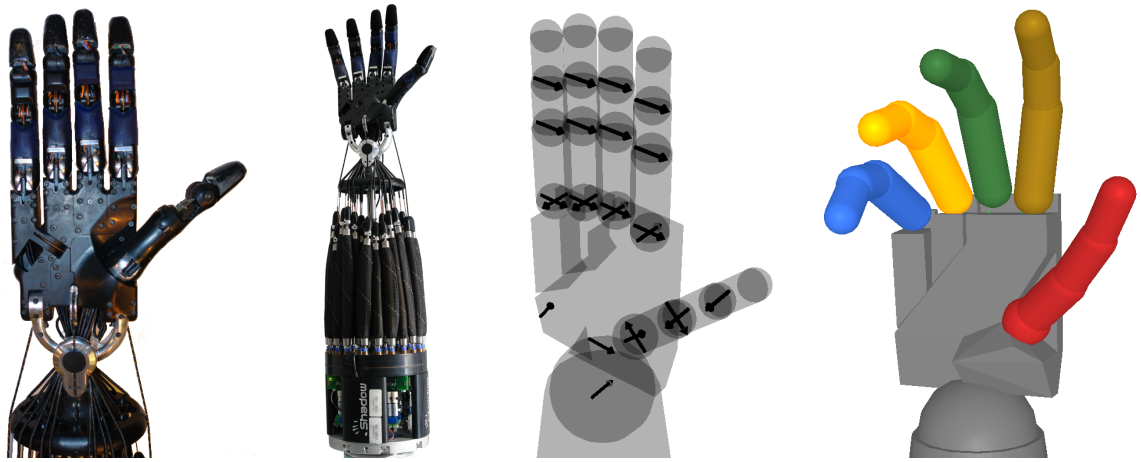
## 3.1. Robotic Hands

From the beginning of robotics on, the term *hand* in the context of a robotic system has been used for the end-effector parts of robot arms which directly interact with the environment. Whereas this comparison of such robotic manipulator systems is somehow natural and seems to be very appropriate at first sight, most of the 'robotic hands' do not have much in common with their human archetype after which they are named. Indeed, most of the robotic end-effectors used in industry are highly specialised for performing one very specific task and the design is kept as simple as possible often only consisting of a two- or three-yaw gripper or a pincer.

Whereas such end-effectors completely lack the dexterity and versatility of human hands, the development of human-like multi-fingered dextrous hands has received much attention in the robotics research for the last few decades. Here, one of the most important far reaching goals is to bring robots into the human every-day life where the robot helper is envisioned for example to assist the human in the household and in elderly care or nursing. Related to this goal, it is save to say that one of the key challenges that have to be tackled is the development of an anthropomorphic dextrous robotic hand. Most importantly, it constitutes an enormously versatile and dextrous tool which is able to manage the huge variety of tasks occurring in human every-day life. It may seem that this alone may not necessitate an anthropomorphic, but only a sufficiently dextrous hand design. However, as the human environment mostly is optimised for the use of humans and thus for the use with human hands, it becomes more clear that a robot requires a very human-like hand in order to be able to operate in such environments.

Indeed, during the last decades, researchers and engineers have made huge advances in constructing and building anthropomorphic dextrous robot hands. A nice overview of the history of the development of various hand designs can be found in (Murray et al., 1994). Under the most important milestones in the development of anthropomorphic dextrous hands are for example the Salisbury (or Stanford/JPL) Hand (Salisbury, 1985), the Utah/MIT Hand (Jacobsen et al., 1986), the DLR II Hand (Butterfass et al., 2001), the Gifu Hand (Mouri et al., 2002), and the Shadow Dextrous Hand (Reichel & The Shadow Robot Company, 2004).

In parallel, some non-anthropomorphic dextrous robot hands have been developed mostly



(a) (b) (c) (d)

**Figure 3.2.:** The Shadow Dextrous Hand. **(a)** Close-up of the hand. **(b)** Hand together with the forearm holding the artificial muscles. **(c)** Visualisation of the hand kinematics (black arrows correspond to the axes of hinge joints) **(d)** The simulation model.

for industrial use. However, since they are less complex to control and often more robust and more reliable to use, some of them have also been widely used in research. Examples are the three-fingered BarrettHand[1] with four degrees of freedom (DOF) and the more recent Schunk Dextrous Hand[2] with seven DOF.

In the following, the Shadow Dextrous Hand and the Gifu Hand III will be described in more detail since these two are the robotic platforms which are mentioned later on in this thesis.

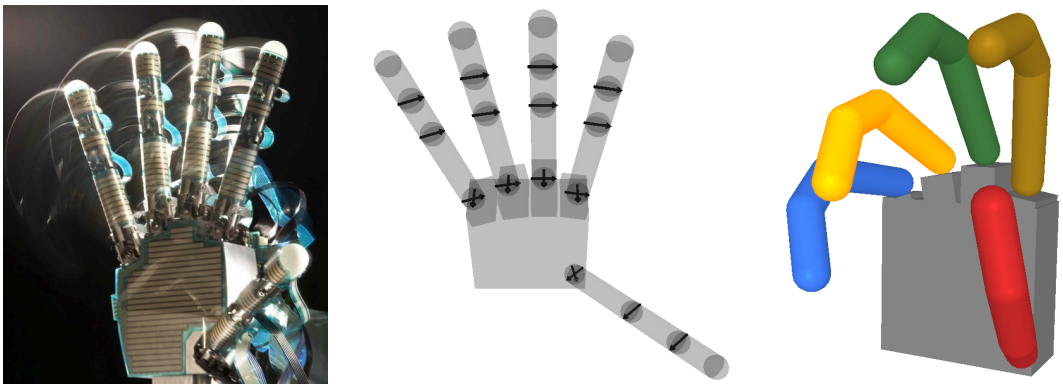
### 3.1.1. The Shadow Dextrous Hand

The Shadow Dextrous Hand (Figure 3.2; (Reichel & The Shadow Robot Company, 2004)) from the Shadow Robot Company Limited, London, UK – in the following *ShadowHand* – is a five-fingered anthropomorphic robot hand system with 20 DOF. The self-contained system consisting of the hand and a forearm has a weight of approximately $4kg$ and a size which is comparable with a large human hand/forearm. The hand design tries to capture the natural human movement abilities providing similar joint limits as the corresponding human joints (see Figure 3.2(c)) and a conditional coupling of the middle and distal phalanges. However, due to the complexity of the human thumb anatomy (e.g. (Hollister et al., 1992, 1995)), the thumb design had to be widely simplified consisting only of hinge joints whose axes are all orthogonal or parallel to each other and do intersect. Here, especially the two proximal base joints feature different kinematic characteristics than the human pendants.

The actuation of the joints is realised by antagonistic artificial muscle pairs for each DOF which are made of rubber and driven by air pressure. Due to the compressibility of the air and the elasticity of the rubber, the fingers provide a certain intrinsic compliance.

Whereas the thumb provides five DOF realised by five independent joints, the fingers each provide three DOF with four joints since the middle and distal joints are coupled in

[1] http://www.barrett.com
[2] http://www.schunk.de



(a) (b) (c)

**Figure 3.3.:** The Gifu Hand III. **(a)** Close-up of the hand. **(b)** Visualisation of the hand kinematics (black arrows correspond to the axes of hinge joints) **(c)** The simulation model.

a human-like manner such that the angle of the distal joint is larger than or equal to the middle joint angle. The wrist provides two DOF for pan and tilt movements and another DOF in the palm imitates the human ability to bow the palm plane.

Inspired by the human model, the muscles are mounted on the forearm and are connected with the corresponding joints via tendons.

In the current version of the hand available at the Bielefeld University, Bielefeld, Germany, each finger including the thumb provides 34 tactile sensors in the finger tip. The other finger segments as well as the palm do not have tactile sensors.

This hand design constitutes the target hand model for the data glove mapping presented in Section 3.3. Further on, due to the mapping from the glove-related human hand model onto the ShadowHand, the hand postures described in Section 5.1 to 5.5 can be directly realised this hand.

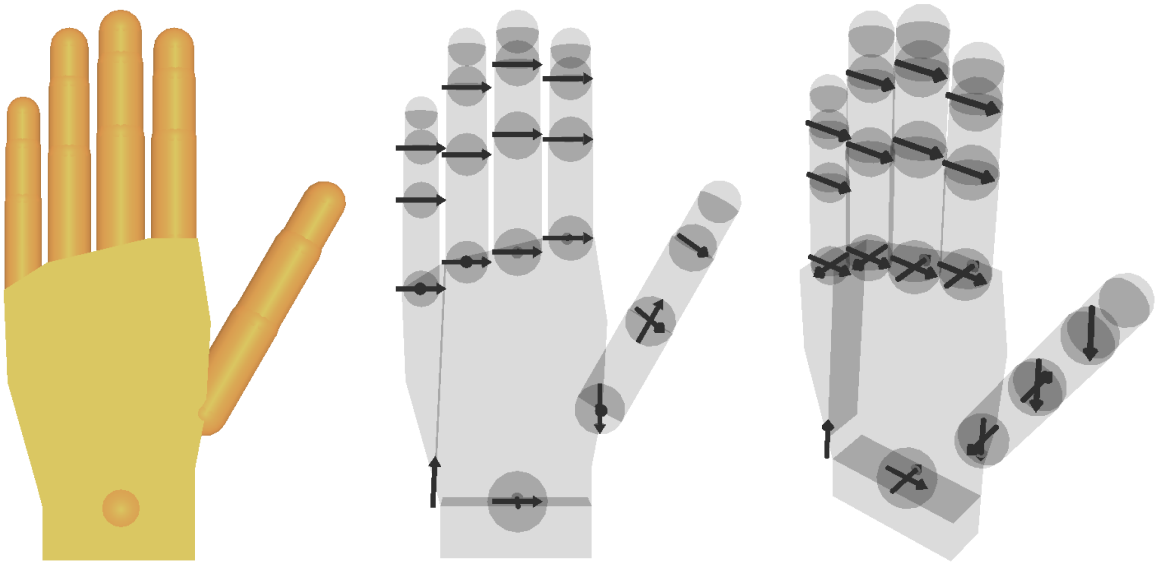
### 3.1.2. The Gifu Hand III

The Gifu Hand III (Figure 3.3; (Mouri et al., 2002)) from the Dainichi Company Limited, Gifu, Japan – in the following *Gifu Hand* – is a five-fingered anthropomorphic robot hand with 16 DOF. The self-contained system consisting only of the hand (without forearm) has a weight of only $1.4kg$ and a size slightly larger than the ShadowHand (only the hand; see previous section). As shown in Figure 3.3(b), the kinematics of the hand is comparable to the ShadowHand whereas no wrist and no palm joint is available.

In contrast to the ShadowHand, the actuation of the joints is realised by small servomotors inside the fingers enabling a design without forearm or external actuation devices. Whereas the servomotors enable a precise and fast control, they do not inhere any compliance. In the Gifu Hand, the thumb provides four DOF realised by four independent joints and each finger provides another three DOF with four joints since the middle and distal joints are coupled in a fixed manner such that both joints have the same angles.

In the current version of the hand available at ATR, Kyoto, Japan, no tactile sensors are available.

This hand design has been used for the implementation of the closed-loop manipulation control presented in Section 5.5.



(a) (b) (c)

**Figure 3.4.:** The human hand model. **(a)** Visualisation in simulation. **(b-c)** The kinematic model: the finger segments/links are depicted in gray whereas the hinge joint axes are visualised as black arrows.

## 3.2. The Human Hand Model

The *human hand model* (see Figure 3.4) has been designed as a basis for the representation of CyberGlove sensor readings in the form of a joint angle vector. This vector describes the corresponding hand posture including the wrist. In order to keep the representation simple and close to the design of the data glove, the model exclusively consists of hinge joints. The complex kinematics of the thumb (Hollister et al., 1992, 1995) is especially simplified: all joint axes are orthogonal or parallel to each other and do intersect.

In this form, most of the 22 bend sensors in the glove can directly be associated with a single joint in the kinematic structure of the hand model as visualised in Figure 3.4(b-c). Exceptions to this direct correspondence are the finger abduction / adduction joints, for which only indirect sensors are featured by the CyberGlove design (see next section for details), and the additional rotational axis (joint) in the thumb. This extra joint enables the model to approximate the passive side movement of the metacarpophalangeal joint of the thumb (Hollister et al., 1995) which cannot be measured by the CyberGlove, but constitutes an important issue for opposing the thumb to the other fingers.

Another simplification has been made in order to include the palm arching in the model. Since the arching itself could not be simulated, it has been replaced by a simple hinge joint in the palm plane (see Figure 3.4(b-c)), similar to the ShadowHand design.

In general, the orientations of the joint axes are chosen such that rotation in positive sense around the axes corresponds to moving the connected segments towards the inner of the hand. In the case of the flexion joints, this corresponds to closing the fingers towards the palm. For the abduction / adduction joints in the fingers and the thumb, this means that the little and ring finger positively adduct in the direction of the thumb, the middle and first finger positively adduct in the direction of the little finger and the thumb positively adduct in the direction of the first finger (see Figure 3.4(b-c)).

## 3.3. From Cyberglove to ShadowHand: A Two-Part Mapping

The desired way of using the CyberGlove to control the robot hand or to generate corresponding training data was already outlined in the introduction: the aim is to imitate the demonstrated hand posture as a whole. In the case of the ShadowHand, this corresponds to mapping the 22 CyberGlove sensor readings onto the corresponding 24 joint angle values of the hand such that the resulting posture of the ShadowHand resembles the hand posture demonstrated by the CyberGlove.

However, instead of realising a direct mapping from the CyberGlove to the ShadowHand, the human hand model described in the previous section is used as an intermediate level. The design of this intermediate model reflects the structure of the data glove. Its joints and joint groups intuitively correspond to angles measured by the glove sensors and therefore to the human hand wearing the glove. Although the ShadowHand itself implements such intuitive correspondence in most of its joints, the two thumb base joints require a more specific and complex mapping (see Section 3.5). Because of this "thumb anomaly", it is desirable to split up the overall mapping into two partial mappings. The first one maps from the raw glove sensory readings onto the joint values of a general *human hand model* representing the mapping from the glove to the human hand. The second one maps the new hand model onto the ShadowHand representing the mapping from the human hand to the robot. In this manner, the first general mapping can be used as a basis for further specific mappings onto different (robot) hand models.

## 3.4. First Mapping: From Cyberglove to Human Hand Model

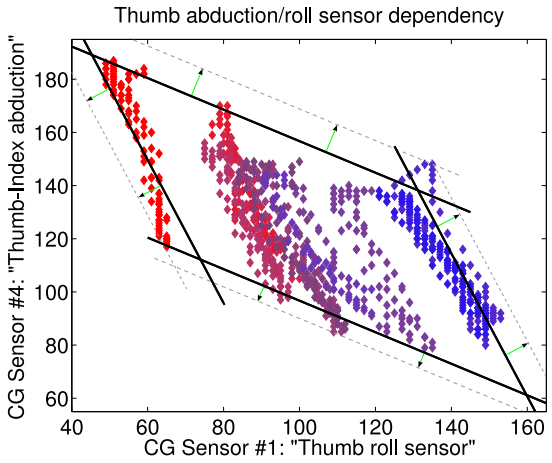
In general, the design of the CyberGlove (Figure 3.1) and the placement of most of the bend sensors on the glove allow for a simple linear mapping between minimal and maximal sensor readings and the corresponding minimal and maximal finger joint values. Two exceptions to this simple case render the generation of an overall mapping a less trivial task: On the one hand, the readings of the finger spread joint sensors do not only depend on the spreading degree of the two corresponding fingers, but also heavily depend on the finger flexion of the neighbouring fingers (see Figures 3.1(b-c)). On the other hand, the thumb abduction and roll sensors mutually influence each other (see next section).

In addition, a general shortcoming of the sensor placements needs to be handled in a specific way: the glove design features only three spread sensors for the four fingers. The corresponding absolute joint positions therefore cannot be fully specified.

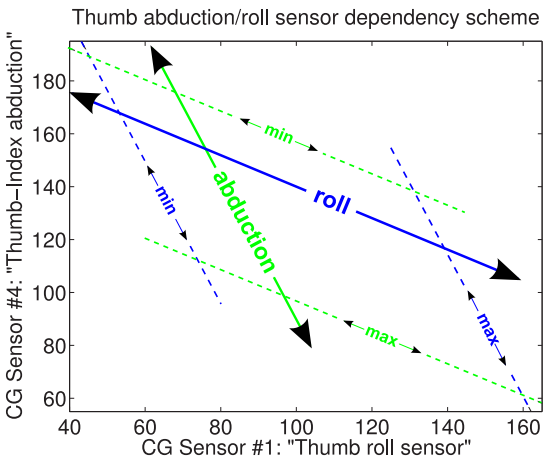
Contrary to Turner (2001) where this problem is bypassed by assuming the middle finger abduction/adduction joint to be fixed, the presented approach aims at realising a model in which all fingers can perform abduction/adduction movements. As replacement for the required constraint, the first and little finger are assumed to be always spread at the same angle. While this new assumption may not significantly improve mapping errors, it indeed enables the model to capture side-movements of the middle finger and therefore enhances its flexibility. These side-movements are important for the turning of a bottle cap, which will be investigated in a later chapter.

### 3.4.1. The thumb abduction / roll sensor mapping.

There are two bend sensors provided in the CyberGlove in order to measure the thumb abduction and roll angles: the "Thumb roll sensor" (CyberGlove sensor no. 1) is placed




(a) Recorded thumb data from abduction movements keeping the roll angle fixed (different colours correspond to different roll angles). The solid black lines represent approximated data borders ("approximatively bounding" parallelogram), the green arrows illustrate the calibration process according to Method 2 (see text for details) and the dashed grey line depict the corresponding calibration result.




(b) Model of the abduction/roll sensor dependency extracted from the data recordings shown in (a). The "approximatively bounding" parallelogram determines the minimal and maximal value borders (outliers get cropped). It can be interpreted as non-orthogonal roll/abduction angle frame in sensor space. The origin is situated in the upper-left corner of the parallelogram, the roll angle axis in sensor space corresponds to the upper border line and the abduction axis to the left border.

**Figure 3.5.:** Data-driven thumb mapping model: visualisation of the thumb roll/abduction dependency ("Thumb-Index abduction", sensor #4) and roll sensor ("Thumb roll sensor", sensor #1).

inside of the top layer and the "Thumb-Index abduction sensor" (CyberGlove sensor no. 4) is placed on top of the glove between thumb and first finger, similar to the three finger spread sensors (see Figure 3.1). Contrary to what their names suggest, they do not explicitly measure only one effect, but are correlated and depend on each other. However, since the two sensors are not interfered with movements other than thumb rolling or abduction, a simple mapping from the two sensor values onto the two roll and abduction angles can be generated.

As basis for such mapping, a data set from the "Thumb roll sensor" and the "Thumb-Index abduction sensor" were recorded in ten different roll positions from 0° to 90° with constant step size of 10°. In each step, the thumb was moved from the minimal to the maximal abduction limit. Figure 3.5(a) depicts the resulting 1116 sensor readings, where the marker colours encode the corresponding roll angles from 0° (bright red) to 90° (bright blue).

An approximative mapping can be constructed by fitting the data points into an "approximatively bounding" parallelogram as shown in Figure 3.5: the sides of the parallelogram can be interpreted as non-orthogonal axes of an askew compressed frame. Projecting the sensor values onto these axes yields relative positions in the valid value ranges for the rolling and abduction of the thumb (outliers get cropped). The absolute angle values can be obtained by linearly interpolating between the minimal and maximal values. Alternatively,

minimum and maximum borders and the current sensor values can be projected onto the sensor axes using the askew axes as projection directions. This yields intersection points $x_0^{min}, x_0^{curr}, x_0^{max}$ on the horizontal "Thumb roll sensor" axis and $y_0^{min}, y_0^{curr}, y_0^{max}$ on the vertical "Thumb-Index abduction sensor" axis. Therefore, the thumb abduction angle, $\alpha$, and the thumb roll angle, $\beta$, can be denoted as:

$$\alpha = \frac{y_0^{min} - y_0^{curr}}{y_0^{min} - y_0^{max}} \cdot (\alpha_{max} - \alpha_{min}) \tag{3.1}$$

$$\beta = \frac{x_0^{curr} - x_0^{min}}{x_0^{max} - x_0^{min}} \cdot (\beta_{max} - \beta_{min}) \tag{3.2}$$

where $\alpha_{min}, \alpha_{max}$ and $\beta_{min}, \beta_{max}$ denote the minimal and maximal values for the thumb abduction and roll angle, respectively.

### 3.4.2. Calibration of the thumb mapping

In principle, two main ways of using this model are useful. They differ in their flexibility at handling user-specific hand variations or a change in the glove sensor structure (e.g., due to the loss of elasticity of the glove):

1. The first method does not need any user-specific model adaptation and is therefore very fast and easy to use. The data is recorded as described above and the corresponding values for $x_0^{min}, x_0^{max}$ and $y_0^{min}, y_0^{max}$ are calculated. The values can be saved and used as fixed glove-specific calibration for all experiments and all users.
   Whereas this method does not yield an optimal mapping result, it allows for the CyberGlove to be used without any further calibration. The mapping accuracy is sufficient for many situations. In addition, by using this fixed calibration as a default configuration, it can be combined with the next model calibration.

2. The second method uses the initial sensor recordings as an example for the general dependency and behaviour of the sensors. A general trend is reflected in the form of the approximatively bounding parallelogram shown in Figure 3.5 and can be described by the gradients of its bordering lines. Instead of keeping the values $x_0^{min}, x_0^{max}, y_0^{min}$ and $y_0^{max}$ fixed (as in method 1), only the gradients $m$ and $y$-axis intercepts $b$ of the bordering lines are fixed. Since the gradients $m_1$ of the left and $m_3$ of the right border are equal as well as the gradient $m_2$ of the top and $m_4$ of the bottom border lines, they are combined to $m_{1/3}$ and $m_{2/4}$, respectively. The corresponding $y$-axis intercepts $b_1$ to $b_4$ have to be stored separately.
   A calibration of this model can be performed by moving the thumb to its extremal abduction and roll positions and replacing the bordering lines of the parallelogram such that it completely bounds the recorded data. Figure 3.5(a) visualises such calibration process (green arrows) and its calibration result (dashed grey lines).
   One drawback of this method is the susceptibility to data outliers resulting from input noise or badly performed hand movements (e.g., the user can use his other hand to move the thumb to extreme positions yielding unnatural limits or erroneous sensor readings when the other hand directly pushes onto one of the bend sensors).
   In practice, this way of using and calibrating the CyberGlove has proven to be very fast, comfortable and yields quite accurate sensor mappings when performed thoroughly. Therefore, this has become the standard procedure for calibration which has been used for all the CyberGlove data utilised in this thesis.

### 3.4.3. The finger spread sensor mapping.

The CyberGlove provides three bend sensors to measure finger abductions (see Figure 3.1): sensor #11 ("Index-Middle abduction sensor"), sensor #15 ("Middle-Ring abduction sensor") and sensor #19 ("Ring-Pinky abduction sensor"). The "Thumb-Index abduction sensor" (#4), in contrast to its misleading name, is placed on top of the middle thumb joint and connects the thumb to the palm instead of the index finger. Whereas this way of sensor placement is beneficial for the mapping of the thumb joints because the thumb can be handled independently from the other fingers (see previous paragraph), it cannot be used to improve the mapping of the finger spread joints.

As shown in Figure 3.1, the finger abduction sensors are placed on top of the glove, each between the associated fingers. In the zero-spread position of the fingers, the sensor strips are fully bended; spreading then results in less bending.

Since each of these sensors is connected to two fingers and therefore describes the spread angles of two joints, each single sensor measures the angle *between* the fingers rather than the two angles of the underlying finger spread joints. Furthermore, since only three of such sensors are provided in the CyberGlove design to describe the angles of four spread joints in the hand, the sensor-to-angle mapping is under-specified, that is, no absolute specification of the angle values can be made without a further condition. To this end, Turner (2001) assumes the middle finger to be fixated in the zero-spreading position which provides an absolute landmark at which the relative finger positions can be oriented. Whereas Turner (2001) reports accurate mapping results for the evaluated movements, an important drawback of this method is that no spread movements of the middle finger can be recorded.

Since the aim of this approach is to use the CyberGlove as an input device for dexterous manipulation tasks and the spread movements of the middle finger can provide important effects for the whole manipulation movement, a different way to resolve the problem of the under-specified sensor mapping was pursued. The little and index fingers are assumed to be coupled such that they are always spread equally. This condition as well does not hold for any possible finger movement. However, it constrains the movement space of the CyberGlove less severely than completely disabling the spreading of the middle finger.

The computation of this constraint is straightforward. The absolute sensor values $s_i$ (whereas $i = 1$ corresponds to the sensor between forefinger (FF) and middle finger (MF), $i = 2$ to the one between MF and ring finger (RF), and $i = 3$ to the one between RF and little finger (LF)) of the three sensors are expressed in terms of a relative value $x_i$ compared to the value range:

$$x_i = \frac{s_i^{max} - s_i}{s_i^{max} - s_i^{min}} \in [0; 1]. \tag{3.3}$$

Since the sensors are fully bended in the zero-spread position, the maximal sensor value $s_i^{max}$ corresponds to the minimal spreading. Vice versa, $s_i^{min}$ corresponds to the maximal spreading.

The relative overall finger spreading $\gamma$ can be denoted as

$$\gamma = \frac{\sum x_i - \sum x_i^{min}}{\sum x_i^{max} - \sum x_i^{min}} = \frac{1}{3} \sum x_i \tag{3.4}$$

since $x_i^{min} = x^{min} = 0$ and $x_i^{max} = x^{max} = 1$. The angular value relative to the minimal spreading can be interpolated between the empirically determined minimal and maximal overall finger spreading angles (i.e. $\phi_{min}^{FF/LF} = -11°$ and $\phi_{max}^{FF/LF} = 60°$):

$$\phi^{FF/LF} = \gamma(\phi_{max}^{FF/LF} - \phi_{min}^{FF/LF}) \tag{3.5}$$

and be split up into the appropriate proportions $\phi_i, (i = 1 \dots 3)$ corresponding to the three inter-finger sensors:

$$\phi_i = \frac{x_i}{\sum x_j} \phi^{FF/LF}. \tag{3.6}$$

Finally, the absolute joint angles $\Theta$ are computed starting with the outer fingers (FF and LF) to realise the constraint. On the basis of these values, the two inner fingers (MF and RF) can be set according to the $\phi_i$ proportioning (in the case of the real ShadowHand and its model and the Human Hand model, the joint axes directions of the finger spread joints are such that positive rotation moves the fingers to the inside of the hand. The first and little finger therefore move towards the ring and middle finger)[3]. The equations can be denoted as follows:

$$\Theta_{FF} = \Theta_{FF}^{min} + \gamma(\Theta_{FF}^{max} - \Theta_{FF}^{min}) \tag{3.7}$$

$$\Theta_{LF} = \Theta_{LF}^{min} + \gamma(\Theta_{LF}^{max} - \Theta_{LF}^{min}) \tag{3.8}$$

$$\Theta_{MF} = \Theta_{FF} - \phi_1 - \Theta_{MF}^{min} \tag{3.9}$$

$$\Theta_{RF} = \Theta_{LF} - \phi_3 - \Theta_{RF}^{min} \tag{3.10}$$

### 3.4.4. The abduction sensor dependency

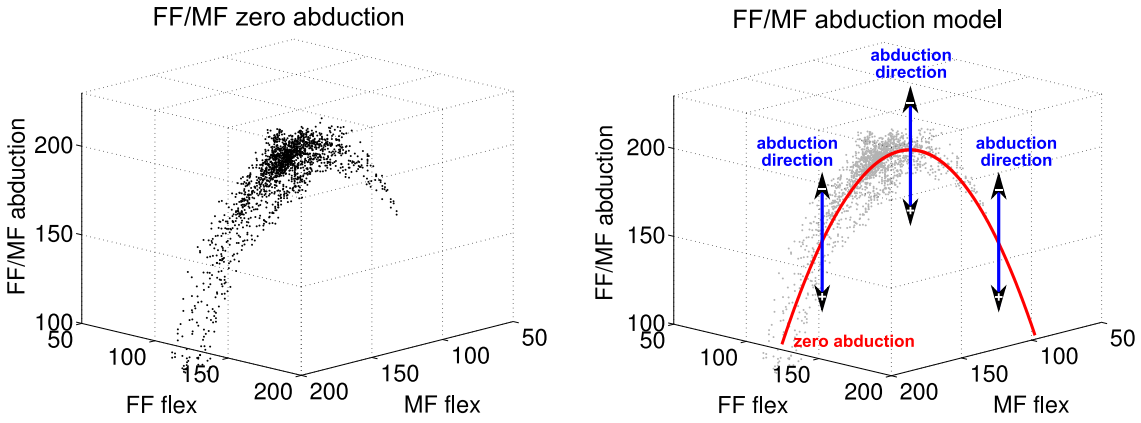
Besides the problem of the under-specified sensor-to-joint mapping of the spread sensors described in the previous paragraph, another intricate difficulty arises from the positioning of the spread sensor strips on top of the glove. Whereas the bend sensors are well suited for the flexion joints since they can be incorporated into the glove in a flat position, the abduction sensor strips need to be placed on edge on top of the glove between the spread joints of the corresponding fingers (see Figure 3.1(a)). Whereas this indeed enables the measurement of finger spreading, the spread sensor values also strongly depend on the (relative) flexion of the neighbouring fingers as shown in Figure 3.1(b-c): when the one finger is flexed to its maximum and its neighbouring finger is not flexed at all while the fingers are not spread (both spread joint angles = 0), then the corresponding spread sensor gets heavily unbent and twisted at the same time. Since the twist cannot be detected by the sensor, the sensor reading as such cannot be distinguished from a finger spreading yielding the same sensor bending.

In order to compensate for this effect on the joint mapping, the correlation between abduction and corresponding neighbouring flexion sensors is evaluated. Figure 3.6(a) shows the dependency of the FF/MF abduction sensor readings on its neighbouring flexion sensors. Depicted are different relative finger flexions for no finger spreading. The abduction angles of both fingers are therefore zero for all the data.

The resulting "data cloud" resembles to a noisy paraboloid-like surface whereas the quadratic shape depends on the difference between the readings of the two flexion sensors. In terms of the corresponding finger postures, this means that the abduction sensor value is maximal when the neighbouring fingers are flexed equally (or more precise, the flex sensor readings are equal) and decreases quadratically with an increasing difference between the two finger flexions.

Evaluating the sensor behaviour for the contrary movement, that is a finger spreading movement with fixed finger flexions, revealed that the flexion sensor readings are not

[3] the actual *closing* direction of the spread joints of the real ShadowHand changed several times resulting in the same changes for its simulation model. For the inverse situation, the correct joint angles can be straightforwardly obtained by multiplication of $\Theta_{\cdot}$ with $-1$.




(a) 'Zero abduction' sensor recordings from the first finger (FF)/middle finger (MF) abduction sensor (#11). Shown is the dependency of those recordings on the neighbouring FF/MF flexion sensors (#5/#8). The FF/MF zero abduction value highly depends on the FF/MF flexion.

(b) Model of the abduction mapping: the FF/MF flexion sensors (#5/#8) dependency of the zero abduction sensor value shown in (a) is modelled as a two-dimensional paraboloid-like surface (red, for the sake of illustration clarity visualised as 1D parable). The 'abduction axis' (blue) however is constant and parallel to the axis of the abduction sensor #11.

**Figure 3.6.:** The abduction mapping model of the fingers. Depicted is the example of the first finger / middle finger sensors (CyberGlove sensors: FF Flex ("Index finger inner joint", glove sensor #5), MF Flex ("Middle finger inner joint", glove sensor #8) and FF/MF Abd. ("Index-Middle abduction", glove sensor #11).

sensitive to abduction movements since they only cause sensor twists. Consequently, the 'abduction direction' in Figure 3.6 is parallel to the vertical "FF/MF abduction" axis as visualised in Figure 3.6(b). Increasing sensor values correspond to decreasing abduction and vice versa (caused by the glove design).

The same characteristics can be found in the corresponding recordings for the MF/RF and RF/LF abduction sensors (see Figure 3.8(a)).

In order to incorporate these observations into the sensor-to-joint mapping, the previously fixed maximal sensor values $s_i^{max}$ are substituted in Equation (3.3) by dynamic zero-abduction values $s_i^{zero}(\cdot)$ which are adapted according to the sensor readings of the corresponding neighbouring flexion joints (see below). This modification slightly changes the conceptual way of handling the abduction sensors: the first version linearly interpolates between minimal and maximal abduction whereas the minimal abduction corresponds to a finger crossing and thus a negative abduction. In the second version, the interpolations are conducted only in the positive abduction range between the dynamic zero-abduction and maximal abduction. Negative abductions corresponding to finger crossings result in negative coefficients $x_i$ in Equation (3.3) and therefore yield extrapolations of the positive abduction range. This modification of the conceptual view point comes with no significant changes in the mapping results, but indeed enables to incorporate the dynamic zero-abduction values.

### 3.4.5. The dynamic zero-abduction model

For the calculation of the dynamic zero-abduction values, a parametrisable model can be defined that inheres the quadratic dependency on the flexion difference and can be fitted to the varying specific characteristics of the different abduction sensors. The final model depending on a height parameter $h$, the two neighbouring flexion sensor readings $s_{\mathrm{l(i)}}$, $s_{\mathrm{r(i)}}$ ($l(i)$: flexion sensor left from abduction sensor $i$; $r(i)$: flexion sensor right from abduction sensor $i$) and a parameter vector $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_6)$ can be denoted as follows:

$$s_{\mathrm{i}}^{zero}\left(s_{\mathrm{l(i)}}, s_{\mathrm{r(i)}}; h, \boldsymbol{\lambda}\right) = h(1 - c \cdot \Delta^2) - d \tag{3.11}$$

where

$$\Delta = \frac{s_{\mathrm{l(i)}} - s_{\mathrm{l(i)}}^{min} - \lambda_3}{s_{\mathrm{l(i)}}^{max} - s_{\mathrm{l(i)}}^{min}} - a \cdot \frac{s_{\mathrm{r(i)}} - s_{\mathrm{r(i)}}^{min}}{s_{\mathrm{r(i)}}^{max} - s_{\mathrm{r(i)}}^{min}} \ , \tag{3.12}$$

$$a = \lambda_1 + \lambda_2 \left(1 - \frac{s_{\mathrm{r(i)}}^{max} - s_{\mathrm{r(i)}}}{s_{\mathrm{r(i)}}^{max} - s_{\mathrm{r(i)}}^{min}}\right) \ , \tag{3.13}$$

$$c = \lambda_4 - \lambda_5 \left(\frac{s_{\mathrm{l(i)}}^{max} - s_{\mathrm{l(i)}}}{s_{\mathrm{l(i)}}^{max} - s_{\mathrm{l(i)}}^{min}} + \frac{s_{\mathrm{r(i)}}^{max} - s_{\mathrm{r(i)}}}{s_{\mathrm{r(i)}}^{max} - s_{\mathrm{r(i)}}^{min}}\right) \quad \text{and} \tag{3.14}$$

$$d = \lambda_6 \cdot \frac{s_{\mathrm{l(i)}}^{max} - s_{\mathrm{l(i)}}}{s_{\mathrm{l(i)}}^{max} - s_{\mathrm{l(i)}}^{min}}. \tag{3.15}$$

The height parameter $h$ of the surface can either be seen as an additional parameter or coupled with the maximal abduction sensor reading $s_{\mathrm{i}}^{max}$.

Figure 3.7 visualises the effects of the model parameters $\lambda_1, \ldots, \lambda_6$ in comparison to the *default* shape (Figure 3.7(a)). The parameters allow for

- rotating ($\lambda_1$), see Figure 3.7(b)
- bending ($\lambda_2$), see Figure 3.7(c) and
- shifting ($\lambda_3$), see Figure 3.7(d)

the parabolic-like surface as well as modifying

- the width ($\lambda_4$), see Figure 3.7(e),
- the back/front closure ($\lambda_5$), see Figure 3.7(f) and
- the back/front tilt of the shape ($\lambda_6$), see Figure 3.7(g).

Figure 3.7(h) depicts an example for using all parameters at the same time.

For the optimisation of the model parameters, least squares can be used. In practice, however, due to the rather noisy data samples, the resulting zero-abduction-surfaces tended towards overfitted solutions with poor generalisation abilities. In addition, the specific characteristics of the mapping problem yielded that it is preferable to overestimate the true zero abduction value than to underestimate it: underestimating favours the mapping on negative finger abductions and therefore – in terms of the resulting hand posture – exaggerated

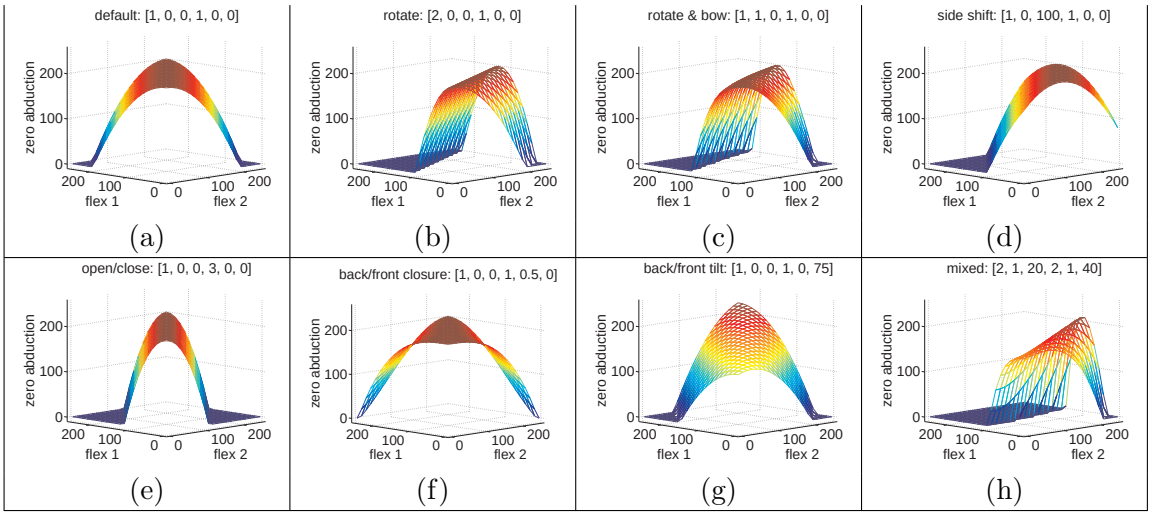



**Figure 3.7.:** Visualisation of the model parameters. **(a)** default configuration $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \lambda_6] = [1, 0, 0, 1, 0, 0]$. The zero-abduction surface has parabolic form and is symmetric in the both flex joints (flex 1, flex2). **(b)** $\lambda_1$: rotation of the isosurface. **(c)** $\lambda_2$: similar to b), but rotation depends on the value of flex2, realising a bow form in addition to the rotation. **(d)** $\lambda_3$: left/right shift. e) $\lambda_4$: open/close the parabolic form. **(f)** $\lambda_5$: change of parabolic form from front to back. **(g)** $\lambda_6$: tilt of the isosurface from front to back. **(h)** an example using all parameters at the same time.

finger crossings. Such crossings look unnatural and can easily cause undesirable finger collisions when actuating the corresponding hand postures on a real robot. The alternative to overestimating the zero level in contrast leads rather to stronger finger abductions which usually look natural and do not yield problems with finger collisions.

While such tendency towards overestimation can be incorporated in the least squares optimisation by modifying the corresponding loss function in an adequate manner, it introduces new (fairly non-robust) meta-parameters in the model.

As the surface and the sample data are naturally embedded in three-dimensional space and therefore allow for an easy visualisation and the model parameters directly correspond to specific surface characteristics (see Figure 3.7), one straightforward alternative to an automatic optimisation is a visually guided manual optimisation of the parameters. Since this evaluation has to be done only once for each abduction sensor of the glove, it does not restrict the practical use of the model.

Figure 3.8 depicts the recorded finger abduction data and the resulting manually optimised zero-abduction surfaces for the three finger abduction sensors of the (first) right-handed CyberGlove II from the Neuroinformatics Group at Bielefeld University. The strategy of favouring an overestimation of the true zero-abduction value is clearly visible in Figure 3.8(c), where data recordings and model surfaces are overlaid. Here, the surfaces represent an upper bound rather than a regression of the recorded data.

### 3.4.6. Calibration of the finger mapping

Since the parameters $\boldsymbol{\lambda}$ of the dynamic zero-abduction model depend in the main on the glove and not on the user who wears it, they only have to be calculated once for each glove. The calibration that is necessary each time the glove is used consists only of the evaluation of the minimal and maximal sensor readings $s_i^{min}$ and $s_i^{max}$ for each joint $i$.

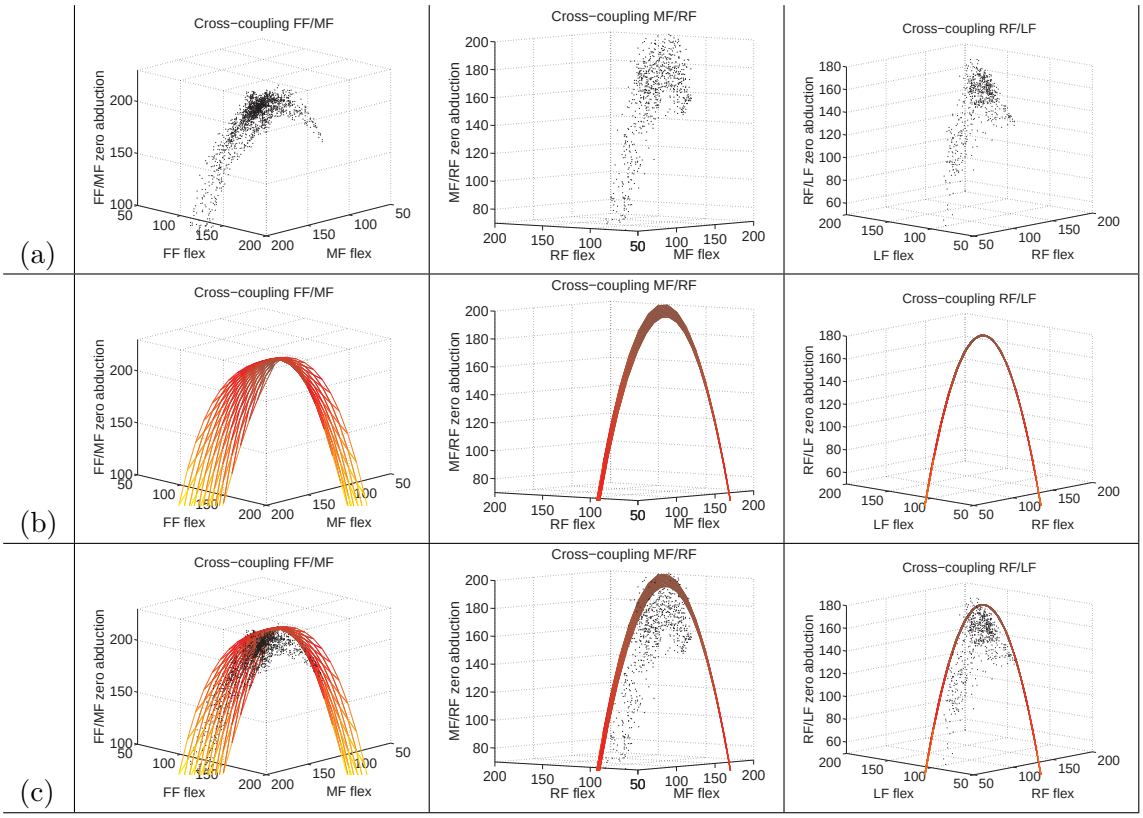



**Figure 3.8.:** Recorded finger abduction data and extracted models. **(a)** abduction sensor readings for zero-abduction depending on the flexion of the neighbouring fingers. left: forefinger (FF) and middle finger (MF), centre: middle finger (MF) and ring finger (RF) and right: ring finger (RF) and little finger (LF). **(b)** the corresponding zero-abduction models after manual optimisation of the presented parameters. **(c)** recorded data with overlaid model.

### 3.4.7. Evaluation of the mapping

For the evaluation of the presented mapping, the required data recordings for the thumb base joints and the finger abduction joints have been performed with the first right-hand CyberGlove of the Neuroinformatics Group at Bielefeld University. On this basis, the fixed mapping parameters have been determined for the thumb mapping according to Method 2 in Section 3.4.2, i.e. the gradients $m_{1/3} = -\frac{78}{29}$ and $m_{2/4} = -\frac{51}{86}$ and the $y$-axis intercepts $b_1 = 249$, $b_2 = 244$, $b_3 = 572$, and $b_4 = 147$ of the bounding parallelogram. For the finger abduction mappings, the parameters $h$ and $\boldsymbol{\lambda}$ have been determined for each abduction sensor according to the dynamic zero-abduction model described in Section 3.4.5. The resulting values are for the FF/MF abduction: $(h = 220, \boldsymbol{\lambda} = (1, 0.3, 22, 1.7, 0.75, 40))$, for the MF/RF abduction: $(h = 200, = (1, 0, 37, 2, 0, 0))$ and for the RF/LF abduction: $(h = 170, \boldsymbol{\lambda} = (1, 0, 6, 2.3, 0, -30))$. The minimal and maximal joint angles of the hand model have been set to adequate values corresponding to the natural joint limits of the human hand. These fixed parameters are stored and constitute the fixed basis for all subsequent calibrations.

The calibration, that has to be performed every time the glove is used, therefore reduces to recording the minimal and maximal readings of the bend sensors of the glove. This can be done by moving the hand to its natural joint limits while wearing the glove.

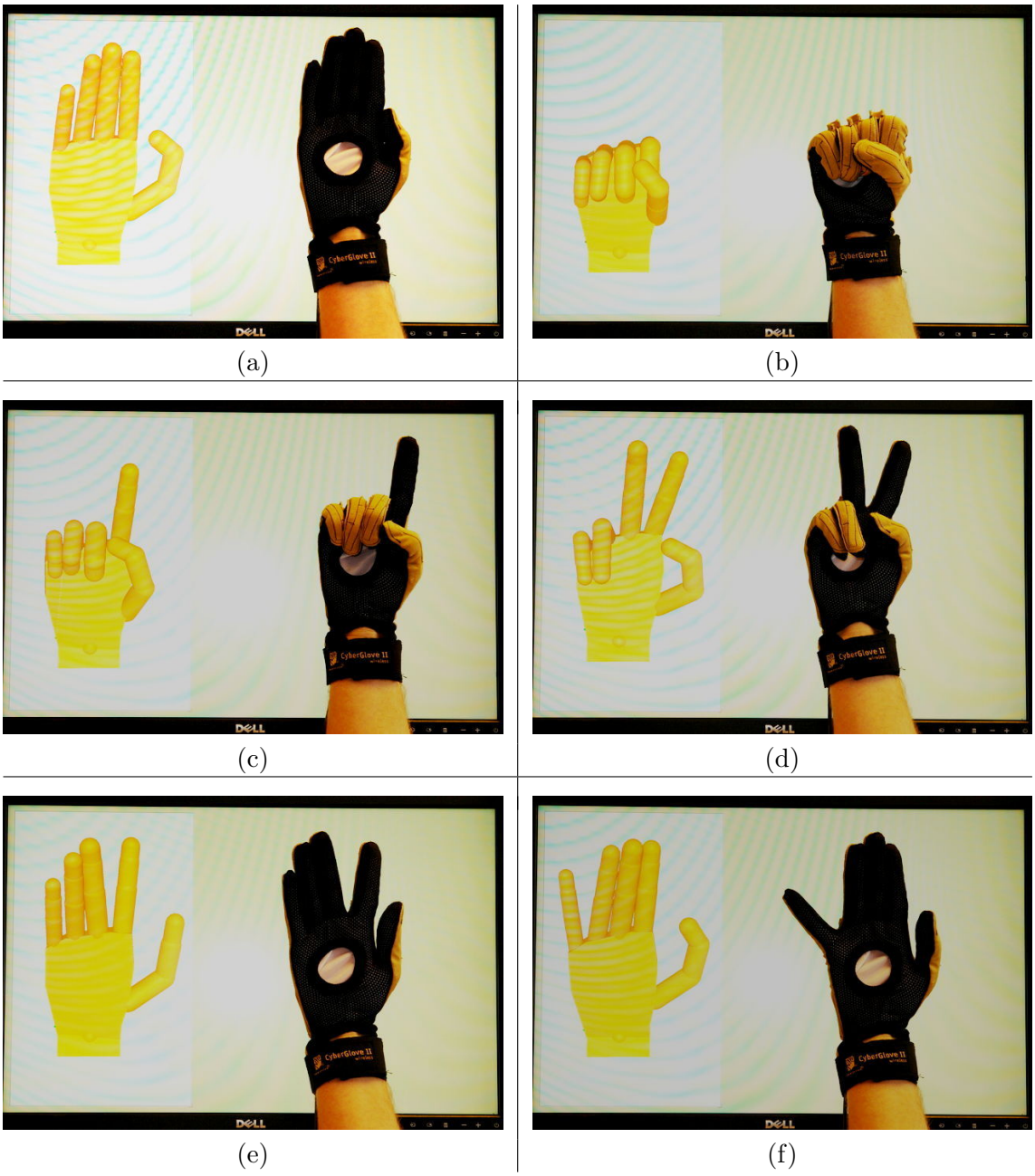



**Figure 3.9.:** Exemplary hand posture mappings as illustration of its visual fidelity (the figure is continued in Figure 3.10).

Figures 3.9 and 3.10 show demonstrated hand postures and the corresponding mapping results. The figures illustrate that the mapping successfully provides a high visual fidelity of the resulting hand postures.
Figures 3.9(a-b) illustrate that simple hand postures which do not involve finger abductions yield faithful mappings.
Figures 3.9(c-d) correspond to situations in which the dynamic zero-abduction model successfully avoids the spreading of the fingers (the abduction sensor between (c) index finger and middle finger or (d) middle finger and ring finger is fully bent here).

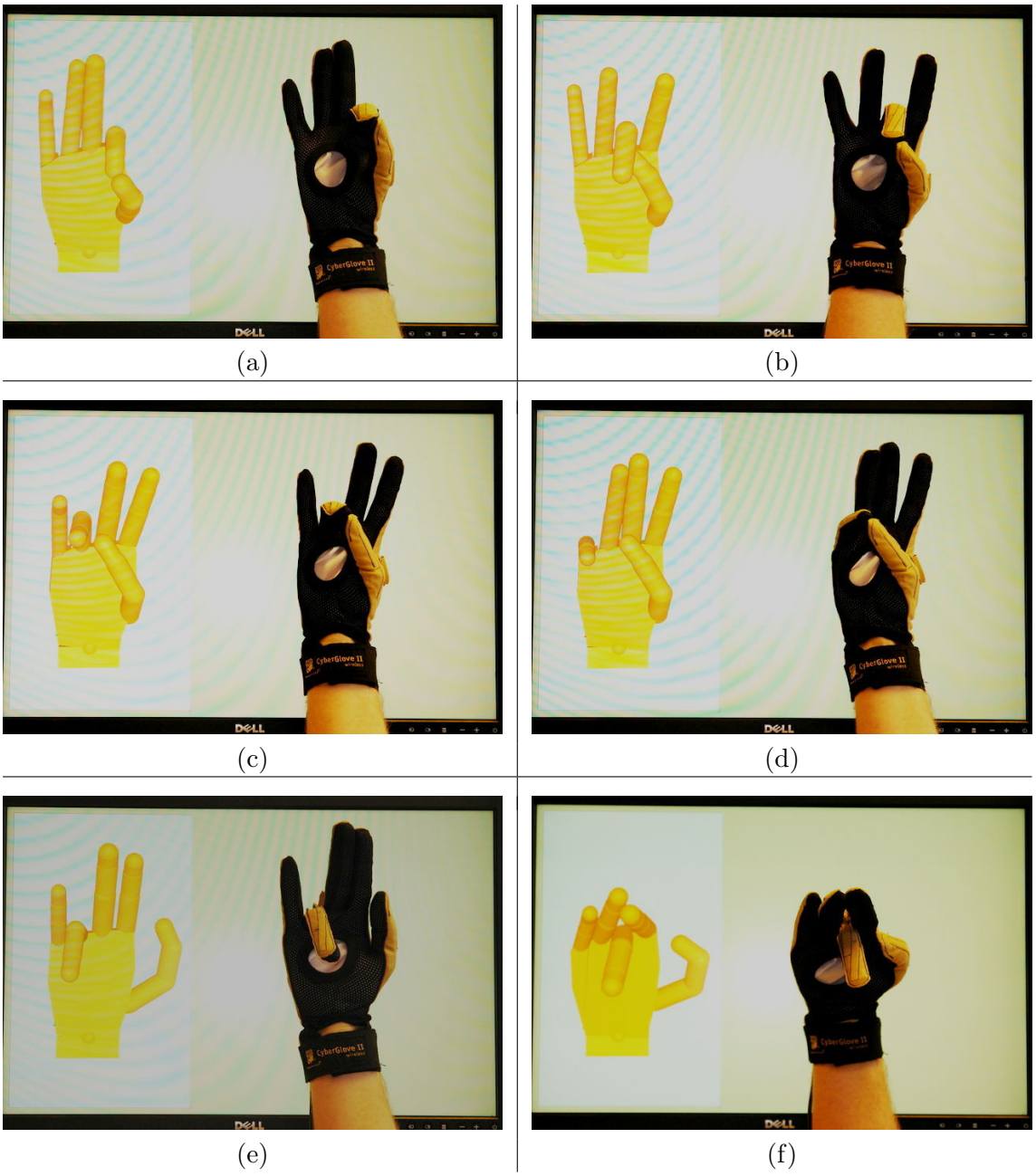
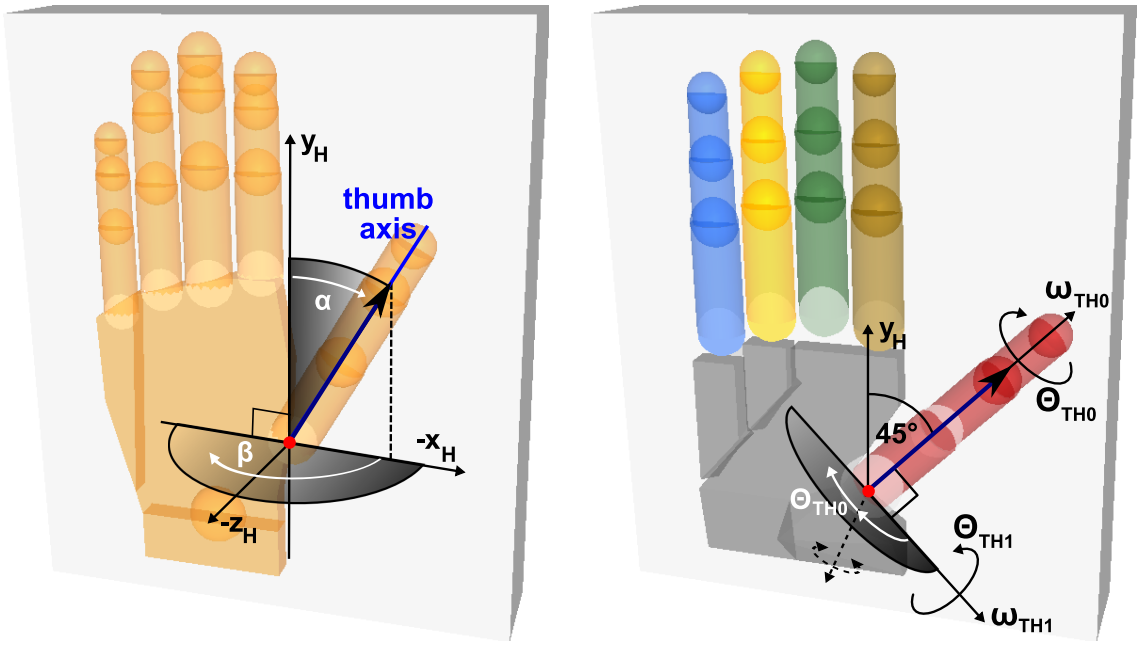



**Figure 3.10.:** Exemplary hand posture mappings as illustration of its visual fidelity (the figure is the continuation of Figure 3.9).

Figures 3.9(e-f) demonstrate the behaviour of the additional constraint for the finger abduction sensors, which had to be included in the mapping in order to overcome the underspecified sensor-to-joint mapping (i.e. that index finger and little finger are always equally spread). Whereas the mapping is still adequate in case (e), only spreading the little finger as shown in (f) results in deviations from the demonstrated posture.
Figures 3.10(a-d) illustrate the performance of the mapping in the cases of the four fingertip/thumb tip contacts. Whereas such contacts can be established for the index (a) and middle (b) finger, the same result cannot be realised for the ring (c) and little (d) finger.




(a) $\alpha/\beta$ representation of the thumb configuration realised in the human hand model ($\alpha$: thumb abduction angle, $\beta$: thumb roll angle).

(b) TH0/TH1 representation of the thumb configuration realised in the Shadow Dextrous Hand (depicted is the kinematically corresponding simulation model).

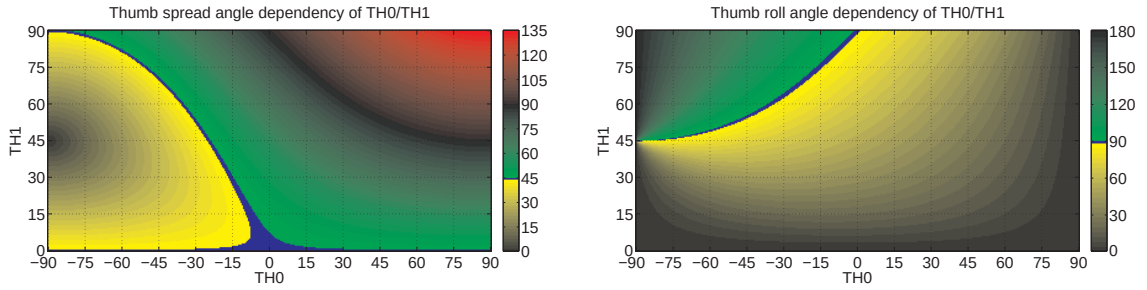
**Figure 3.11.:** The two different thumb configurations from the human hand model and the ShadowHand.

Especially Figure 3.10(d) reveals, that this shortcoming does indeed not result from the mapping, but rather from the glove design and the missing flexibility of the hand model's palm: the figure reveals that the little finger of the demonstrator's hand is moved towards the thumb by extensively arching the hand palm. This movement can however not be represented by the hand model and also not be sufficiently measured by means of the provides palm arch sensor in the glove.
Figure 3.10(e) depicts a hand posture which involves two finger abduction sensors and constitutes an example for the successful combination of two dynamic zero-abduction model.
Figure 3.10(f) demonstrates that the mapping can also successfully cope with finger crossings.

## 3.5. Second Mapping: From Human Hand Model to ShadowHand

The first mapping step described in the previous section realises a mapping from the raw sensor readings from the CyberGlove to a joint angle representation of the Human Hand Model (Section 3.2). The design of this simplified model of a hand corresponds to the human hand that wears the CyberGlove. This first mapping step therefore provides an angular description of the hand posture that is demonstrated by the glove.

The second step of the mapping then closes the gap between the human hand that controls




(a) Thumb *spread* angle depending on thumb joint angles TH0 and TH1. All values in degree.

(b) Thumb *roll* angle depending on thumb joint angles TH0 and TH1. All values in degree.

**Figure 3.12.:** The dependency of the spread and roll angles on the thumb joint angles TH0 and TH1 of the ShadowHand.

the CyberGlove and the robotic hand (or simulated hand model) that is to be controlled.

As mentioned before, in the case of the ShadowHand, most of the joints already correspond to the human hand structure. For these joints, no further mapping is required.

However, due to a rather non-intuitive design of the two proximal thumb hinge joints (meant to provide the functionality of the human proximal thumb ball-socket joint), the corresponding joint angle values from the Human Hand Model cannot be used directly for the ShadowHand. The two different thumb joint configurations are depicted in Figure 3.11: whereas the abduction ($\alpha$) and roll ($\beta$) angles of the thumb are directly realised by joints in the Human Hand Model (see Figure 3.11(a)), the ShadowHand design provides a fixed axis $\omega_{TH0}$ (which does not move with the thumb) and a thumb-relative axis $\omega_{TH1}$ to control the basic thumb orientation, as shown in Figure 3.11(b).

The highly non-linear relationship of $\Theta_0/\Theta_1$- and $\alpha/\beta$-representation is depicted in Figure 3.12 where dependency of the actual spread (a) or roll (b) angle of the thumb on the TH0/TH1 angles is shown. The angle values are encoded in the colour of the corresponding points. The blue regions represent the centre values of the provided spread/roll angle intervals. The singularity of the ShadowHand joint design is clearly visible in the plots: for $\Theta_{TH1} = 0$, that is, the thumb axis corresponds to the joint axis $\omega_{TH0}$, neither the actual spread angle $\alpha = 45°$ nor the actual roll angle $\beta = 0°$ change for the whole range of $\Theta_{TH0}$. Instead, the thumb is rotated around the thumb axis $\omega_{TH0}$.

The mapping from the $\alpha/\beta$- to the $\Theta_{TH0}/\Theta_{TH1}$-representation (and back) can be computed analytically. To this end, the position of a point on the thumb axis with unit distance to the thumb base frame is calculated for the two different representations yielding the following equation system ($\Theta_0 = \Theta_{TH0}, \Theta_1 = \Theta_{TH1}$; see Appendix A.1 for details):

$$\begin{pmatrix} \frac{1}{2}\sqrt{2}\cos(\Theta_1) + \frac{1}{2}\sqrt{2}\sin(\Theta_0)\sin(\Theta_1) \\ \frac{1}{2}\sqrt{2}\cos(\Theta_1) - \frac{1}{2}\sqrt{2}\sin(\Theta_0)\sin(\Theta_1) \\ \cos(\Theta_0)\sin(\Theta_1) \end{pmatrix} \stackrel{!}{=} \begin{pmatrix} \sin(\alpha)\cos(\beta) \\ \cos(\alpha) \\ \sin(\alpha)\sin(\beta) \end{pmatrix} \tag{3.16}$$

Solving this system for $\Theta_0/\Theta_1$ yields:

$$\Theta_0(\alpha,\beta) = \arctan 2\left(\frac{\frac{1}{2}\frac{\sqrt{2}(-\cos(\alpha)+\sin(\alpha)\cos(\beta))}{\sqrt{\frac{1}{2}-\cos(\alpha)\sin(\alpha)\cos(\beta)+\frac{1}{2}\sin(\alpha)^2\sin(\beta)^2}}}{\frac{\sin(\alpha)\sin(\beta)}{\sqrt{\frac{1}{2}-\cos(\alpha)\sin(\alpha)\cos(\beta)+\frac{1}{2}\sin(\alpha)^2\sin(\beta)^2}}}\right). \tag{3.17}$$

$$\Theta_1(\alpha,\beta) = \arctan 2\left(\sqrt{\frac{1}{2}-\cos(\alpha)\sin(\alpha)\cos(\beta)+\frac{1}{2}\sin(\alpha)^2\sin(\beta)^2},\ \frac{1}{2}\sqrt{2}(\cos(\alpha)+\sin(\alpha)\cos(\beta))\right). \tag{3.18}$$

In the inverse direction, solving the system for $\alpha/\beta$ yields:

$$\alpha(\Theta_0,\Theta_1) = \arctan 2\left(\sqrt{\frac{1}{2}+\cos(\Theta_1)\sin(\Theta_0)\sin(\Theta_1)+\frac{1}{2}\sin(\Theta_1)^2\cos(\Theta_0)^2}\ ,\ \frac{1}{2}\sqrt{2}\left(\cos(\Theta_1)-\sin(\Theta_1)\sin(\Theta_0)\right)\right). \tag{3.19}$$

$$\beta(\Theta_0,\Theta_1) = \arctan 2\left(\frac{\cos(\Theta_0)\sin(\Theta_1)}{\sqrt{\frac{1}{2}+\cos(\Theta_1)\sin(\Theta_0)\sin(\Theta_1)+\frac{1}{2}\sin(\Theta_1)^2\cos(\Theta_0)^2}}\ ,\ \frac{1}{2}\frac{\sqrt{2}\left(\cos(\Theta_1)+\sin(\Theta_1)\sin(\Theta_0)\right)}{\sqrt{\frac{1}{2}+\cos(\Theta_1)\sin(\Theta_0)\sin(\Theta_1)+\frac{1}{2}\sin(\Theta_1)^2\cos(\Theta_0)^2}}\right) \tag{3.20}$$

These solutions hold for the whole valid ranges of $\alpha/\beta$ and $\Theta_0/\Theta_1$, respectively.

### 3.5.1. Calibration of the second mapping step

Since this second mapping step only consists of analytically determining the two thumb base joints as described above and copying the rest of the joint values from the Human Hand Model, no calibration or further mappings are necessary.

## 3.6. Data acquisition with the CyberGlove II

After the calibration of the glove mapping, the glove can be used as interface to acquire hand posture data for manipulation tasks. The presented approach does not optimise the position errors of the fingertips which would possibly result in strange or even infeasible hand configurations. It rather follows the goal of 'visual fidelity' yielding more natural looking hand postures. The resulting joint angles can usually not be used directly as training data: manipulation tasks such as turning a bottle cap for a specific cap radius (see Figure 3.13) require contacts with the corresponding object, and therefore consequentially adequate finger tip positions in space are required.

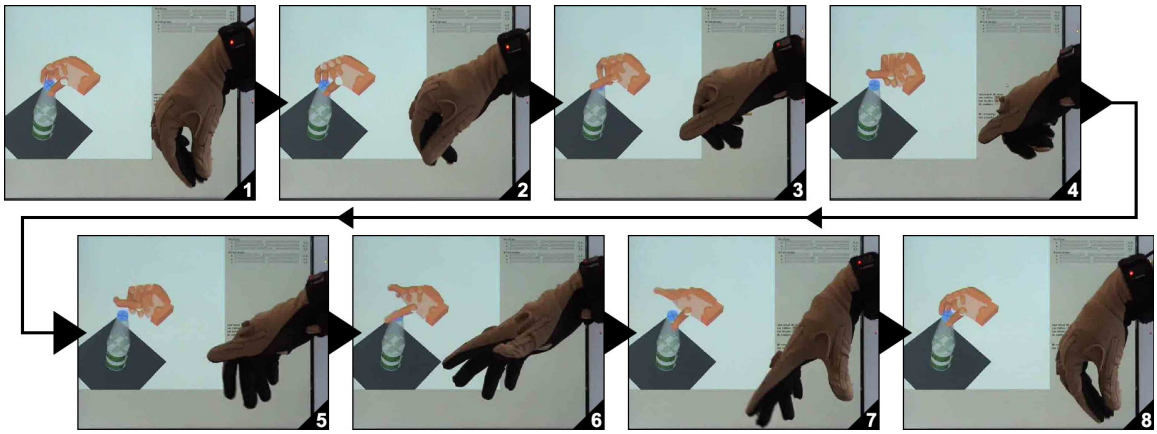



**Figure 3.13.:** Exemplary data acquisition with the CyberGlove II: The glove is used as user interface to control the simulated hand in the corresponding manipulation scene. The targeted manipulation then is performed within the simulation and the training data for manipulation algorithms is recorded from the simulation model – after the corrections of the physics-based simulation.

In order to be able to generate hand posture data that provide both desired features, visual fidelity *and* appropriate finger positions, an additional mechanism can be used which follows the 'Robot Skill Synthesis via Human Learning' paradigm (Oztop et al., 2006). Instead of letting the demonstrator perform the targeted manipulation directly with his own glove-mounted hand, the robot is used as a tool by the human as he performs the task.

While this approach still requires a good mapping from the demonstrator's hand to the robot, a lack of accuracy in either the mapping of the demonstrator's joint angles or in the mapping's ability to handle the structural differences between human and robotic hand, can be compensated for by the human. By allowing the human include this *tool* in his body schema, the human visuo-motor learning capacity can be exploited to generate appropriate training data directly on the robot.

Figure 3.13 depicts an exemplary data acquisition for the manipulation task of turning a bottle cap. The glove is used as an input device to control the hand in the simulated manipulation scene. The manipulation is only performed in the simulation and the human does not have interaction with a real bottle cap. The hand postures, which are controlled by the human through visual feedback and corrected by the collision detection system of the physics-based computer simulation, are recorded from the simulated hand model. Since the data is generated with this (simulated) target robotic platform, the data consequentially is adequate for the robot and can be used to train representations of the corresponding manipulation for it.

## 3.7. Discussion

The Immersion CyberGlove II Wireless is a versatile input device for recording hand posture data. One of its main advantages is that the complete system consists only of the glove and a Bluetooth dongle and is therefore completely portable. In order to take advantage of this portability, it is however necessary to implement an adequate server application including a sufficiently accurate sensor mapping[4].

[4]Immersion ships the glove in a bundle with a server application that is however single-copy-licenced, closed-source and can only be installed on one computer. The last point eliminates the main advantage

Once the server is implemented and the sensory mapping is set up, the glove can be robustly used for grasping and manipulation experiments. In the author's opinion, the method presented in this chapter constitutes a good mapping which provides good accuracy in given limits of the glove's possibilities. If the underlying training data recordings are performed thoroughly, the resulting mapping is able to capture the intrinsic characteristics of the glove design.

In general, CyberGloves (and comparable systems) are useful for capturing hand postures as a whole where the focus lies on natural looking finger postures. The exact positions in space of the fingertips can usually not be extracted perfectly. The main reason for this is that the bend sensors have a limited resolution. In addition to this, although the theoretical range of their sensory values is $[0; 255]$, only a small part of it is exploited in many cases. Also, the way the demonstrator wears the glove slightly differs each time the glove is put and affects the positions of the bend sensors on top of the joints. Such small differences in the positioning can yield small differences in the sensor responses. Since each sensor only measures one joint, the fingertip position is affected by all errors of the joints between fingertip and palm resulting in non-negligible cumulated errors. Furthermore, when mapping the demonstrated hand postures onto a robot hand using a joint angle representation, the design of the robot hand is usually significantly different in size and shape. Actuating a hand posture that corresponds to the demonstrated one according to its general finger postures automatically results in a different fingertip position.

In addition to the above mentioned limitations, the CyberGlove has some intrinsic drawbacks that are caused by its structural design:

- The problems with the finger spread sensors have already been discussed: since only three sensors are provide to measure the spread angles of four fingers, no absolute finger positions can be obtained. This shortcoming could have easily been averted by providing an additional sensor placed at the side of the hand, connecting the palm and the little finger.

- The sensors for the thumb abduction/adduction and the thumb rolling are correlated.

- The behaviour of the palm arch sensor is highly sensitive to the size of the demonstrator's hand. In addition, this sensor only functions correctly in one of three gloves available in the Neuroinformatics Group at the Bielefeld University.

- The behaviour of the wrist sensors is highly sensitive to the size of the demonstrator's hand and to the way the glove is connected to the forearm. Significant changes in the behaviour are also possible during a single demonstration, if the fastener gets loose.

Still, despite some shortcomings of the CyberGlove and the fact that the accuracy of hand posture recordings is not perfect, especially when targeting fingertip positions in space, the following advantages justify its use for tasks of grasping and manipulation:

- It is much cheaper than a professional vision system that yields a higher accuracy.

- It is easily portable (in combination with a self-implemented server).

of its high portability. Since the sensory mapping included is rather poor and in particular the cross-couplings of the spread sensors with the neighbouring flexion sensors are not handle at all, it made sense to implement a new mapping.

- It does not suffer from effects like marker occlusions and marker correspondence problems.
- It does not interfere with other capturing systems (e.g., active cameras of a marker tracking system can interfere with time-of-flight cameras).

In the next chapter, manipulation data for the task of turning a bottle cap has been recorded with a CyberGlove using the joint angle mapping presented here. This was then used to develop and evaluate a new method for the representation of manipulation data.

# 4. Manifold grasping and the road to Structured Manifolds

During the last decades, researchers and engineers have made huge advances in constructing and building anthropomorphic robot hands which have become more and more sophisticated as one can see for example in the Utah/MIT Hand (Jacobsen et al., 1986), the DLR II Hand (Butterfass et al., 2001) and the Shadow Dextrous Hand (Reichel & The Shadow Robot Company, 2004). Together with these developments, researchers are facing the question of how to dexterously control such complex robots with up to 20 degrees of freedom in up to five fingers and a wrist. It quickly became clear that implementing fixed grasp and manipulation programs does not lead to satisfying results as it is very time consuming on the one hand and not robust against or generalisable to differences in the grasping or manipulation situation. Thus, several approaches have been presented to realise more robustness and generalisability.

In the domain of dextrous *grasping*, algorithms bear the challenge to grasp a large variety of objects possibly unknown in shape, weight, and position. To this end, existing approaches nowadays use imaging techniques to estimate the 3D shape and position of the object, or employ tactile sensors with coarse spatial and temporal resolution.

One of the main difficulties is to find grasping strategies and algorithms which generalise to new situations and are therefore sufficiently robust against variations of object position, orientation, or even shape.

In general, there are two opposing approaches to address this issue. The first uses explicit object geometry models to calculate (optimal) contact points on the object surface and subsequently plans a hand posture to realise the contacts and therefore the corresponding grasp. The second does not plan beforehand, but closes the fingers around the object solely based on the tactile feedback of the hand until stable object contact is detected. Most of the approaches of both domains do not accumulate their grasping experience to employ it for future grasping situations.

In the case of "geometry-based" grasping, the algorithm itself inheres the knowledge of how to grasp specific objects, but is not able to match this knowledge dynamically to the presented object. These algorithms only succeed if the geometric representation of the object shape indeed matches the object at hand. If a geometric description of the object is not available in a previously acquired database or cannot be obtained on-line, the object cannot be grasped at all.

On the other side, "contact-based" grasping reacts on tactile events and dynamically adapts the grasping motion to the actual situation. While additional visual feedback may be incorporated, most approaches of this paradigm employ a fixed finger closing strategy, which does not take advantage of an implicit or explicit representation of the object's shape. Hence, these algorithms fail if the initial assumptions about object position and its coarse shape are wrong or do not match to the observed tactile events.

For both of these strategies, several examples exist. Borst et al. (2002) compute a set of optimised contact points based on a detailed geometric object model and calculate a

hand posture that generates the recommended contacts solving a constrained optimisation problem. Later, they relax the need of optimal contact points showing that an average quality grasp performs sufficiently well in most tasks (Borst et al., 2003). Miller et al. (2003) weaken the requirement of a precise geometry information using shape primitives which roughly approximate the object's geometry. Based on these primitives, multiple grasp starting positions and approach vectors are evaluated in simulation using GraspIt! (Miller & Allen, 2000, 2004), and the best one is chosen for execution. Pelossof et al. (2004) combine this approach with support vector machine (SVM) regression to map parametric descriptions of superquadric objects to potential initial hand postures and an associated grasp quality. trained with parameters of superquadric objects, hand parameters of the grasp and the resulting grasp quality, again evaluated with GraspIt!.

The main critique about this domain of approaches is that they are mostly developed in (and for the use in) physics-based computer simulated environments in which object shape and position are precisely known and also used. In real world scenarios for grasping every-day objects, however, such a precise geometrical object model is usually not available. Thus, the precise grasping knowledge cannot be optimally exploited due to the lack of object knowledge. For many of the approaches mentioned above, it is hence doubtable in which extent they can be transfered from the simulation to real world scenarios.

The contact-based grasping, in contrast, requires no object information and is therefore in general better qualified in less perfectly specified grasping situations. Such a control has been presented e.g. by Natale and Torres-Jara (2006) who built a tactile-driven system to explore and grasp objects without prior object knowledge. They conclude that tactile exploration is more powerful in every-day tasks than precise a priori computation. Platt et al. (2002, 2004) treat grasping as an active sensory-driven problem without explicit object knowledge. Multiple control laws are applied whereas subordinate laws operate in the nullspace of super-ordinate ones and hence do not interfere with higher level goals. Steil et al. (2004) and Röthling et al. (2007) proposed an algorithm based on pairs of an initial pre-grasp and a final grasp posture which are selected from a set of common grasps according to human gestures. The grasp is accomplished by a tactile-driven transition from the initial pre-grasp to the final grasp posture, stopping when stable fingertip contacts are detected.

However, without any knowledge about the geometry or shape of the object, but solely based on tactile feedback, all purely contact-based approaches cannot exploit object-specific knowledge to perform expedient task-dependent finger coordination.

In recent years, alternative ways of thinking about grasping have been presented in the field of machine learning that aim at finding less complex representations of the grasping problem. To this end, *eigengrasps* have been taken into account (Ciocarlie et al., 2007) or lower-dimensional manifolds embedded in the hand posture space are used in order to facilitate manual grasp control (Tsoli & Jenkins, 2007).

Also in the early work that has been carried out in preparation of this thesis, such lower-dimensional representations of hand postures have been studied in the context of dextrous grasping (Steffen et al., 2007a, 2007b) with initial work described in (Steffen, 2005). On the one hand, these works present a grasp algorithm that exploits previously collected grasp knowledge for an experience-based and tactile-driven dynamic grasp control. On the other hand, they provide a manifold-based representation of this grasp knowledge together with a detailed evaluation of the characteristics of such manifolds approximated with Self-Organising Maps (SOM, e.g. (Ritter et al., 1992)).

The following section will detail the ideas and insights which came up along with the elaboration of the two corresponding publications, as they constitute an important part of the vision and the motivation on whose basis this thesis has been carried out.

## 4.1. Experience-based Grasping

In (Steffen et al., 2007a), an approach to dextrous robot grasping is described that combines the advantages of geometry-based and contact-based grasping. Using tactile information to infer implicit object knowledge, the algorithm dynamically includes previously acquired grasping knowledge to adapt the grasping motion to the actual situation. The grasping experience is formed by a database of hand postures which previously led to successful grasps. According to observed finger contacts, the most suitable hand posture is selected from the database to guide the grasping process.

The main idea of the grasping algorithm is to augment a tactile-driven grasping heuristics with an *experience base of grasp postures* which can be used to guide the grasping process to promising hand postures. This experience is represented as a database of hand postures $\mathbf{\Theta}$ which previously led to successful grasps of one or more objects in a variety of grasping contexts, i.e. different positions and orientations of the object relative to the hand. Let $I$ be the number of fingers, $N_i$ the most distal joint in finger $i$, then $\mathbf{\Theta}$ denotes the vector $[\Theta_{1,1}, \ldots, \Theta_{1,N_1}, \Theta_{2,1}, \ldots, \Theta_{I,N_I}]^T$ comprising all joint angles of all fingers. It is important to notice that a grasp posture directly corresponds to a specific object and grasping context. Used in another context, the same hand posture might not lead to a successful grasp.

Comprising a set of successful grasp postures, the experience base implicitly provides knowledge about how to grasp the associated object. As counterpart, the tactile information observed during the grasping process provides implicit knowledge of the object shape, position and orientation. A dynamic matching of this context-specific knowledge to the grasping knowledge stored in the experience base yields information about how to grasp the current object in the current situation. Here, only joint angles are utilised for this matching process whose finger segments provide reliable context information. To this end, a *Partial Contact Posture (PCP)* $\mathbf{\Theta}^{pcp}$ is employed specifying only joints between the palm and finger segments having object contact. If $S_{i,j}$ denotes the finger segment directly attached to and moved by joint $j$ of finger $i$, the PCP can be defined more formally as:

$$\mathbf{\Theta}^{pcp} = [\Theta^{pcp}_{1,1}, \ldots, \Theta^{pcp}_{1,N_1}, \Theta^{pcp}_{2,1}, \ldots, \Theta^{pcp}_{I,N_I}]^T \tag{4.1}$$

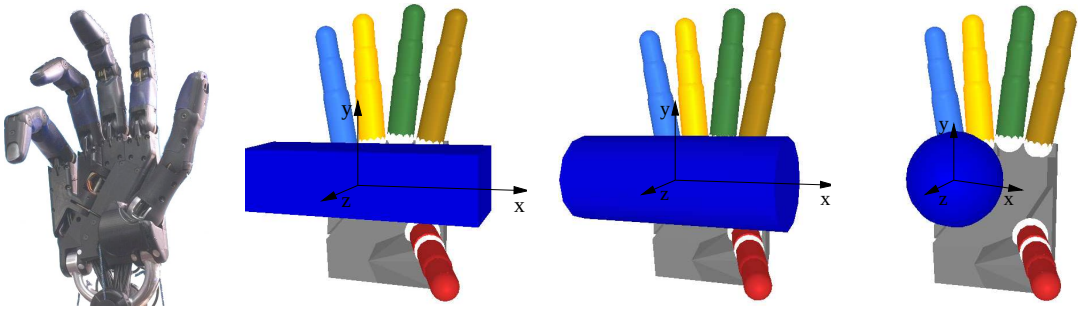
where:

$$\Theta^{pcp}_{i,j} = \begin{cases} \Theta_{i,j} & \text{if a segment } S_{i,(k\geq j)} \text{ has contacts} \\ \text{not specified} & \text{otherwise.} \end{cases} \tag{4.2}$$

Based on this PCP, a modified Euclidean norm $d^{pcp}$ can be used to match a current hand posture $\mathbf{\Theta}^{pcp}$ to the best matching posture $\mathbf{\Theta}^{xp,\star}$ in the experience base $\{\mathbf{\Theta}^{xp}\}$ by minimising $d^{pcp}$, only taking reliable joints into account:

$$d^{pcp}(\mathbf{\Theta}^{xp}, \mathbf{\Theta}^{pcp};\ \mathbf{s}) = \sum_{i,j} s_{i,j}(\Theta^{xp}_{i,j} - \Theta^{pcp}_{i,j})^2 \tag{4.3}$$

Here, $\mathbf{s}$ with $s_{i,j} \in \{0, 1\}$ is used to select the *specified* dimensions of $\mathbf{\Theta}^{pcp}$ for comparison and disable the non-specified ones. Since the PCP requires contact information, a precondition for the experience-based control is the existence of at least one contact. Furthermore, as




(a) (b) (c) (d)

**Figure 4.1.:** The Shadow Dextrous Hand **(a)** and its simulation model **(b-d)**. Depicted is the initial hand posture with the three studied grasp objects: **(b)** box (4×4×16cm), **(c)** cylinder (l:16cm, ∅:6cm), and **(d)** sphere (∅:6cm).

the experience base is a discrete set of hand postures which additionally is affected by noise, a subsequent generic finger closing heuristic similar to the algorithm presented by Röthling et al. (2007) is necessary to establish stable object contacts.
Summarised, there are four different control phases of the algorithm:

(A) Actuating an initial hand posture,

(B) Establishing a first contact in an already experience-influenced process,

(C) Performing the experience-based grasp control and

(D) Applying a generic finger closing heuristic (embedded in phases B and C).

The whole grasp control is embedded in an action-perception-loop which allows a dynamic adaptation to the current grasping context. A schematic overview of the control algorithm is given in Figure 4.2. A detailed description of each step is given below. Single loop iterations will be referred to as *control cycles*.

**(A) Initial hand posture (Pregrasp) (Figure 4.2, box A)** While the purely contact-based grasping algorithm presented by Röthling et al. (2007) is based on a static transition from an initial to a final hand posture, the experience-based approach dynamically selects the currently best fitting hand postures from the experience base to finally reach a grasp posture optimally suited for the actual situation. Nevertheless, the grasping result depends on the initial hand posture which actually pre-determines the first contact occurrence and therefore primes the remaining grasping process. Hence, the *pre-grasp* posture can be selected context-dependent, e.g. incorporating task knowledge or vision results. However, experiments indicate that the algorithm is relatively robust with respect to the initial hand posture such that an empirically evaluated hand posture (see Figure 4.1(c-d)) has been used for all evaluation experiments.

**(B) Establishment of the first contact (Figure 4.2, box B)** Prior to the first contact, the best-match search based on Equation (4.3) cannot be applied because no meaningful PCP is available. Nevertheless, the experience base contains valuable information about typical grasp postures. Hence, during this initial phase, the posture $\mathbf{\Theta}^{xp,\star}$ which best resembles the current posture considering *all* joint angles is actuated: $\mathbf{\Theta}^{pcp} = \mathbf{\Theta}$ with $s_{i,j} = 1$ for all

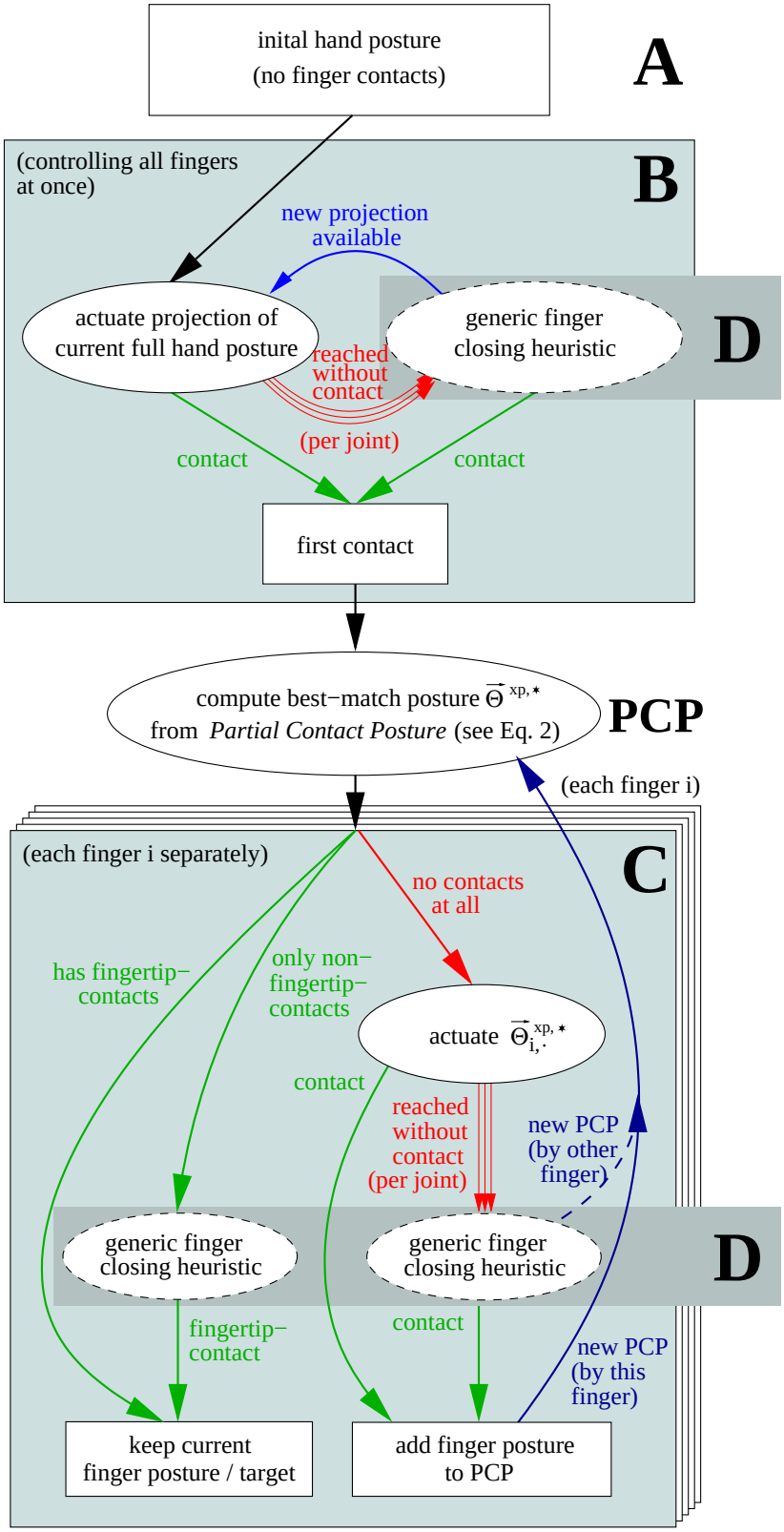



**Figure 4.2.:** Overview of the grasp control algorithm.

joints $j$ of all fingers $i$. If no contact can be evoked in this manner, the generic finger closing heuristic closes the fingers further until another best-match posture $\mathbf{\Theta}^{xp,\star}$ is found and the process can start over. This cycle is repeated until the first contact arises.

**(C) Experience-based control (Figure 4.2, box C)** Together with the first contact, the experience-based grasp control can be initiated by considering the current tactile information to determine the best matching posture $\mathbf{\Theta}^{xp,\star}$ from the experience base:

$$\mathbf{\Theta}^{xp,\star}(\mathbf{\Theta}^{pcp}) = \arg\min_{\mathbf{\Theta}^{xp}} \mathrm{d}^{pcp}(\mathbf{\Theta}^{xp}, \mathbf{\Theta}^{pcp}). \tag{4.4}$$

$\mathbf{\Theta}^{xp,\star}$ then is used as target posture for further motion generation. In order to obtain a stable grasp which additionally allows for flexible manipulation of the grasped object, it is desirable to have stable fingertip contacts for all fingers. Such fingertip contacts ensure that all degrees of freedom can be exploited for manipulation and guarantee at the same time the highest spatial resolution for tactile sensing. According to Figure 4.2(box C), each finger is controlled separately depending on its particular contact state:

1. The finger has no contact(s).
   In this case, the corresponding finger joints are actuated towards the sub-target posture $\mathbf{\Theta}^{xp,\star}_{i,\cdot}$ for finger $i$. If the resulting motion leads to a contact, the PCP changes, a new best-match posture can be computed, and the control proceeds in state 2) or 3), respectively. If the targeted joint angles are reached without evoking any contacts, the generic finger closing heuristic is applied to further close the finger joints.

2. The finger has only non-fingertip contact(s).
   Since the former grasp motion did not lead to fingertip contacts, all joints more distal than the contact segment(s) will be subsequently closed further applying the generic finger closing heuristic again. More proximal joints (corresponding to the specified dimensions of the PCP) maintain their current positions to prevent pushing the object away. In this state, the finger control is no longer affected by updates of $\mathbf{\Theta}^{xp,\star}$ as long as at least one finger contact remains. This prevents backward movements potentially resulting from new target postures and therefore ensures a coherent grasping advance.

3. The finger has fingertip contact(s)
   The grasping objective is accomplished with respect to the corresponding finger and thus, the motion of the finger is halted until the fingertip contact is lost, e.g. due to object motion. In that case, the finger control returns to state 1) or 2) in the next cycle.

If the contact situation changes due to finger motions, the PCP is newly determined and a new best-match posture is selected from the experience database as indicated by the bottom-up arrow in Figure 4.2(box C). Hence during the grasping progress, the targeted grasp posture is always dynamically adapted to the actual context inferred implicitly from the tactile sensory feedback.

**(D) Generic finger closing heuristic (Figure 4.2, box D)** If the grasping experience does not cover the current grasping situation, the described control scheme possibly leads to no fingertip contacts at all – even if the targeted posture $\mathbf{\Theta}^{\mathbf{xp},\star}$ is reached. For example, if the current object is smaller than all previously grasped objects or is shifted in an unusual way.

In this case, a modification of the closing heuristic presented by Röthling et al. (2007)) can be applied without taking the experience base into account. Here, the aim is to continue the finger closing motion until contacts are observed. The main issue to be handled in this phase is the degree of finger flexion determined by the relative closing speeds of the proximal with respect to the distal finger joints, denoted by the ratio $\lambda$. A value $\lambda > 1$ corresponds to a marginal, a value $\lambda \approx 1$ to a distinct flexion. The finger joints continue closing at joint-specific speeds $\dot{\Theta}_{i,j}$ according to the following equations:

$$\dot{\Theta}_{i,j} = \begin{cases} 0 & \text{if a segment } S_{i,(k \geq j)} \text{ has contacts} \\ \gamma_{i,j} & \text{otherwise} \end{cases} \tag{4.5}$$

$$\text{where} \qquad \gamma_{i,j} = \begin{cases} \lambda \cdot \gamma & \text{if } (i,j) \text{ is a proximal joint} \\ \gamma & \text{otherwise} \end{cases} \tag{4.6}$$

The value $\gamma$ specifies the motion speed which is assumed to be positive in order to generate a closing motion. Assuming an adequate experience base with appropriate grasp postures, the generic finger closing heuristic is only applied to compensate for small deviations of the current grasping situation. In the opposite case, having a poorly matching experience base, the grasp motion is mainly controlled by the purely tactile-driven heuristic.

## 4.2. The Grasp Manifold

Grasping experience was introduced in the previous section as a set or a database of grasp postures. However, for appropriately covering a wide range of possible grasping situations, such a database necessitates to comprise a multitude of corresponding hand postures, which increases both the computational costs and the required storage capacity.

Thus, in order to provide such kind of grasping experience in a large scale, a more compact representation of it is necessary. On the basis of the assumption that the valid grasp postures for at least one object form a low-dimensional smooth manifold $\mathcal{GM}$ embedded in the hand posture space, an approach to approximating such a *Grasp Manifold* as compact representation is presented by (Steffen et al., 2007a). The following section describes the method along with an evaluation of the resulting grasping performances.

### 4.2.1. SOM Grasp Manifolds

One powerful and adaptive realisation of such an approximation is the Self-Organising Map (SOM; e.g. (Ritter et al., 1992)). In terms of the *Grasp Manifold*, it consists of a $m$-dimensional lattice $A$ of nodes labelled with a spatial lattice index $\mathbf{a} \in A$ and having attached a reference vector $\mathbf{w_a}$ in hand posture space. A projection of a hand posture $\mathbf{\Theta}$ onto the SOM is determined as the reference vector of the "best-match node":

$$\mathbf{w_{a^\star}}(\mathbf{\Theta}) = \arg\min_{\mathbf{w_a}} \parallel \mathbf{w_a} - \mathbf{\Theta} \parallel . \tag{4.7}$$

Adaptation of the nodes is realised in an iterative procedure similar to vector quantisation methods. Given a training set $\{\mathbf{\Theta}_i\}$ of grasp postures, the learning rule takes the form:

$$\forall \mathbf{\Theta}_i : \forall \mathbf{a}_j : \Delta \mathbf{w}_{\mathbf{a}_j}(t) = \varepsilon(t) \cdot h_{\mathbf{a}_j, \mathbf{a}^\star(\mathbf{\Theta}_i)} \cdot (\mathbf{\Theta}_i - \mathbf{w}_{\mathbf{a}_j}) \tag{4.8}$$

where $t = 1..T$ is the training epoch (cycle over all data points), $\varepsilon(t) \in [0,1]$ is a slowly decreasing learning rate and $h_{\mathbf{a}_j, \mathbf{a}^\star(\mathbf{\Theta}_i)}$ is a Gaussian neighbourhood function diminishing

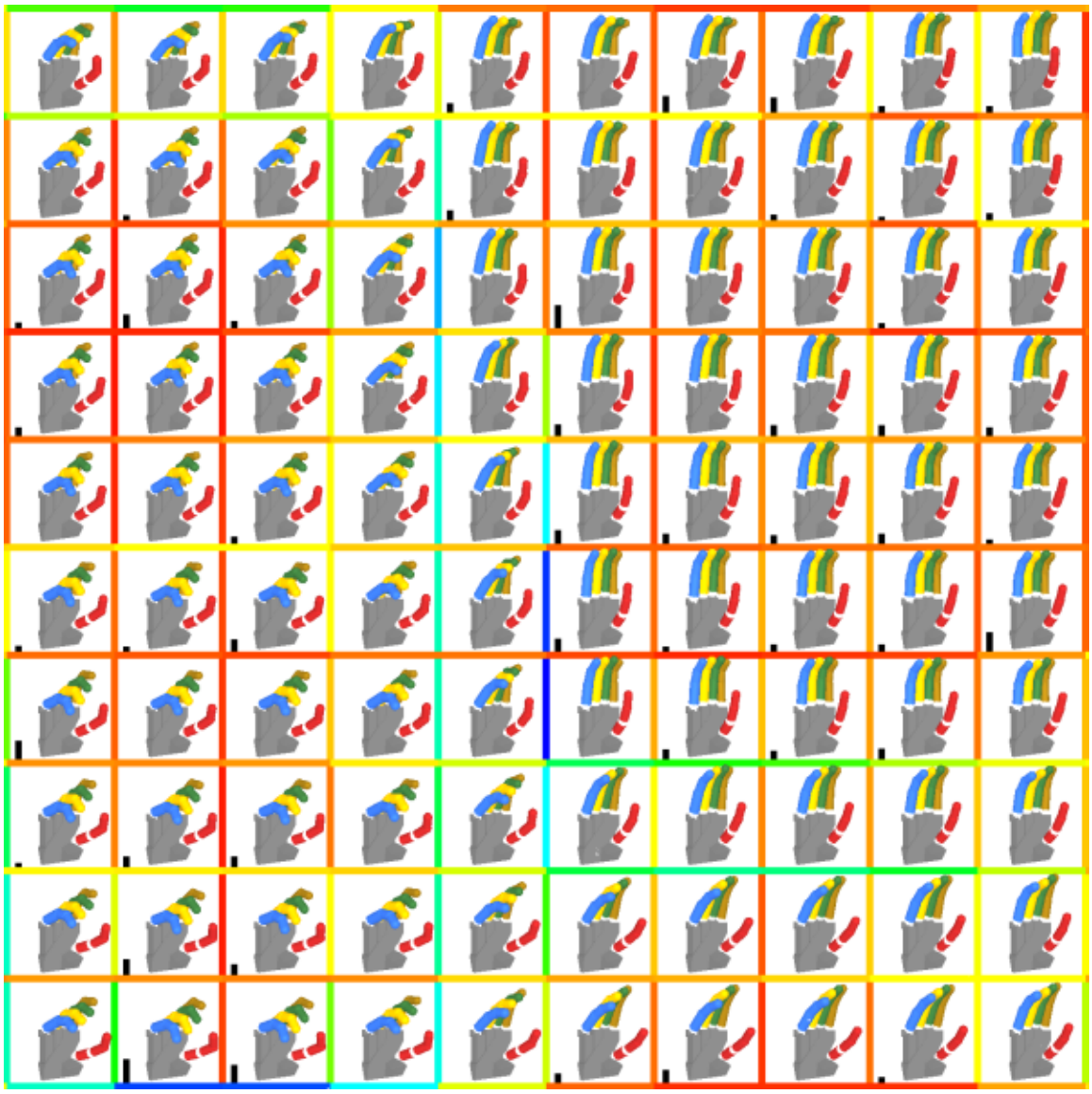


**Figure 4.3.:** 10x10 extract of a cylinder-specific 25x25 SOM *Grasp Manifold*. Each hand posture visualises the reference vector $\mathbf{w_a}$ of one SOM-node. The smooth changing of the postures supports the assumption of the *Grasp Manifold*. The colour of the sub-figure borders encode the inter-node distance (red: minimal distance, green: medium distance and blue: maximal distance). Black bars left from the hand indicate the relative amount of training data that support the corresponding SOM node; no bars indicate the lack of supporting training data.

the learning rate depending on the distance of node $\mathbf{a}_j$ to the best-match node $\mathbf{a}^\star(\mathbf{\Theta}_i)$ in the lattice $A$:

$$h_{\mathbf{a},\mathbf{a}^\star} = \exp\left(-\frac{\| \mathbf{a} - \mathbf{a}^\star \|^2}{2\sigma(t)^2}\right). \tag{4.9}$$

Due to this neighbourhood adaptation, the SOM can learn a smooth *Grasp Manifold* which preserves the topology of the original hand posture space.

Once successfully trained, such a SOM Grasp Manifold represents the set of hand configurations in the hand posture space that is of special interest in the corresponding grasping

context. The task of grasping then can be described as modifying the current hand configuration in an appropriate way such that it converges to a *grasp configuration* in the manifold. The SOM as discrete approximation of such manifold provides efficient means to accomplish this task. By projecting the current hand configuration onto the SOM by performing the best-match search (4.7), the closest grasp configuration in the set of SOM reference vectors can be obtained. In addition, the SOM provides an elegant way to recover grasp postures based on incomplete grasp data – which corresponds to the idea of the partial contact posture (PCP) described in Section 4.1 – and thus to pull incomplete grasp configurations onto the manifold. This mechanism of an "associative completion" was introduced first by Walter (1996) in the context of the Parametrised Self-Organising Map (PSOM, (Ritter, 1993)). It can be realised by replacing the Euclidean norm by the distance metric $\mathrm{d}^{pcp}(\cdot)$ in Equation (4.8) according to the PCP distance in Equation (4.3):

$$\mathrm{d}^{pcp}(\mathbf{x}, \mathbf{x}';\ \mathbf{s}) = \sum_i s_i (x_i - x'_i)^2. \tag{4.10}$$

$\mathbf{s}$ determines the weightings of the single data dimensions. Setting $s_i > 0$ considers the component/dimension $i$ for the projection and a value of $s_i = 0$ deactivates the corresponding dimension. Hence, by setting the $s_i = 1$ for those dimensions $i$ of the current hand posture that already coincide with a grasp posture and setting $s_i = 0$ otherwise, the PCP part of the hand posture can be *activated* for the SOM best-match search and the rest *deactivated*. The SOM best-match search (4.7) for associative completion then can be denoted as

$$\mathbf{w_{a^\star}}(\mathbf{\Theta}) = \arg\min_{\mathbf{w_a}} \mathrm{d}^{pcp}\left(\mathbf{w_a}, \mathbf{\Theta};\ \mathbf{s}\right). \tag{4.11}$$

In this form, Equation (4.11) however corresponds to the search for the best matching posture in the experience base (4.4) used to recover a final grasp posture from a PCP in Section 4.1. Thus, the SOM with the modified distance metric (4.10) for the best-match search implements the same projection onto the experience base as the PCP matching, but for the manifold representation of the experience. This extension of the best-match search comes with no additional computational costs.

Figure 4.3 depicts an extract of a two-dimensional SOM-based *Grasp Manifold* trained on 4220 cylinder grasp postures. Each hand picture represents the reference vector $\mathbf{w_a}$ of a particular SOM node. The visualisation of the SOM shows that similar hand postures are grouped together and change smoothly to other postures which supports the initial assumption of a smooth low-dimensional *Grasp Manifold*. Nevertheless, there are less smooth areas which will be interpreted in the next section.

### 4.2.2. Properties of the SOM Grasp Manifold

For the experiments described in the following, object-specific 25x25 SOMs were trained with a set of grasp postures for one specific object shape. Hence, the particular SOMs are only adequate for grasping objects of the matching type, requiring a prior classification of the object. This has the advantage, that the extension of the system by new objects does not change the behaviour for already learned objects. In principle, however, representing all grasp postures corresponding to various objects *within a single SOM* is possible as well (Steffen, 2005).

Due to the lack of more appropriate input devices (like e.g. data gloves) at the time of the development of this method, the training data for the SOM training was generated

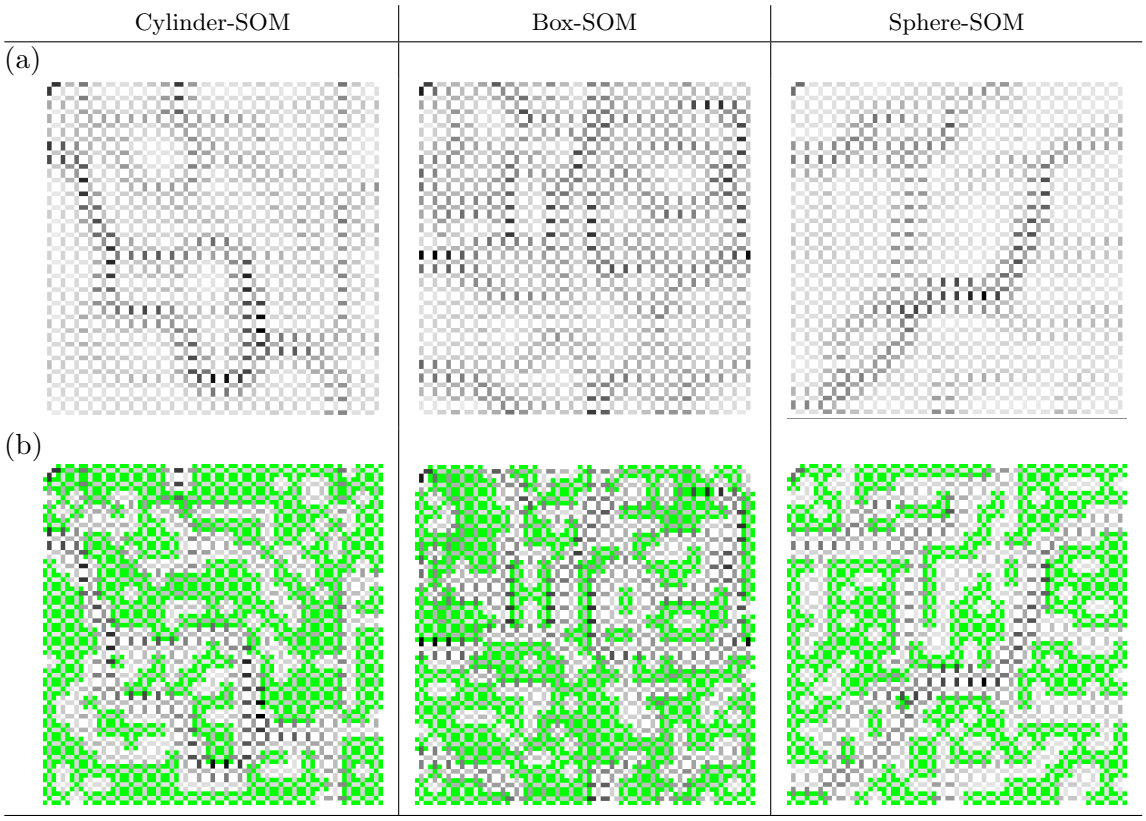
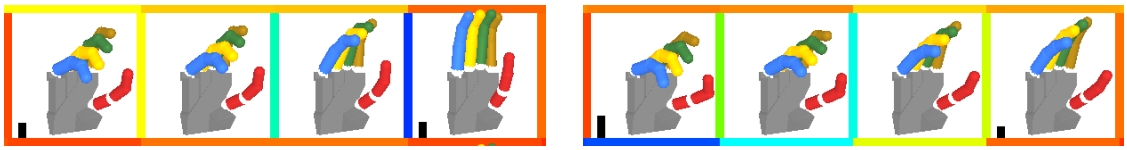



**Figure 4.4.:** Visualisation of the inter-node distances and the data-projection-nodes of the trained 25x25 cylinder-, box- and sphere-SOM. **(a)** inter-node distance structure: The small white squares mark the node positions in the lattice $A$, the large white squares are background. Black node connections denote maximal, light gray connections denote minimal inter-node distances. This corresponds to minimal/maximal mean joint angle errors of $5.9 \cdot 10^{-4}$ deg /2.88 deg (cylinder), $2.50 \cdot 10^{-4}$ deg /2.88 deg (box) and 3.18 deg $\cdot 10^{-4}$ deg /3.01 deg (sphere) degrees, respectively. **(b)** overlaid node matches (green background) when projecting the training data set onto the SOM. The nodes on the cluster borders are only roughly supported by training data.

in physics-based computer simulation. In order to obtain postures with hand/object contacts at every fingertip, the fingertips were connected to the corresponding object. In the simulation environment, virtual springs served to pull the fingers in the direction of the object. The object itself is fixed and by moving the hand manually in a variety of relative hand/object positions and orientations, a total of 10.174 postures were generated, 4.069 for the box, 4.220 for the cylinder and 1.885 for the sphere. Since the springs are only able to establish contacts for small deviations of an initially manually actuated five-fingertip-contacts posture, several of such "starting postures" had to be used. While this method allows for generating a large amount of data representing different regions of the hand posture space, it results in man-made clusters of data around these starting postures.

To train the object-specific SOM-*Grasp Manifolds*, the grasp postures associated with the corresponding object were presented within all 300 learning epochs according to a random distribution (learning rate $\varepsilon(t)$ in Equation 4.8 decreasing from 0.95 to 0.05, standard deviation $\sigma(t)$ of the Gaussian neighbourhood function (4.9) decreasing from 6 to 0.7). The magnification effect of the SOM learning results in a higher density of nodes in the clusters of training data, while there remain some nodes in the space in between. Onto these nodes,



(a) (b)

**Figure 4.5.:** Two 1x4 extracts of Figure 4.3. In both cases (a) and (b), the leftmost and rightmost nodes are supported by training data (indicated by the black bars in the picture left from the hand). The middle nodes do not have data support and represent inter-cluster nodes. Though not supported by training data, they represent useful grasp postures and interpolate nicely between adjacent cluster nodes.

no training data is mapped at all and their distance to neighbouring reference vectors is much larger than on average. Hence, the clusters of the training data and cluster borders are clearly visible (see Figure 4.3 and 4.4(a)). Figure 4.3 is a 10x10 extract of a node visualisation of a cylinder-specific 25x25 SOM. Each hand picture represents the reference vector of the corresponding SOM node. The clusters correspond to areas with similar grasp postures, cluster borders can be identified by highly differing postures from one node to the next. The colours of the sub figure borders describe the inter-node distances: bright red denotes the minimal inter-node distance in the SOM, green a medium and bright blue the maximal distance. A more distinct picture of the inter-node distance structure of the whole SOMs is depicted in Figure 4.4(a) where only the inter-node connections are shown. High inter-node distances are represented by black connections (maximal mean distances per joint: 2.88 deg for the cylinder, 2.88 deg for the box, 3.01 deg for the sphere) and small distances by very light gray connections (minimal distances: $5.9 \cdot 10^{-4}$ deg for the cylinder, $2.50 \cdot 10^{-4}$ deg for the box and $3.18 \cdot 10^{-4}$ deg). Thus, the black "lines" represent the cluster borders where the reference vectors have high inter-node distances.

In Figure 4.4(b), in addition to Figure 4.4(a), the nodes which are supported by training data are marked with a green background. In contrast to the nodes in data clusters, the nodes on the cluster borders are mainly not supported. However, as shown in Figure 4.5, these inter-cluster nodes represent meaningful grasp postures as well and interpolate nicely between adjacent cluster nodes. Thus, the SOM training achieves to learn meaningful bridges between the clusters, representing intermediate grasp postures.

Interestingly, during the application of the experience-based grasping algorithm (keeping the SOM fixed), those inter-cluster nodes are more often winners than the cluster nodes used during the training (see Figure 4.6). This suggests, that during the testing, much more grasp situations are discovered than in training. By resuming the learning phase of the SOMs with the grasp postures generated in testing and performing few learning epochs with small learning parameters (10 epochs, $\varepsilon(t)$ decreasing from 0.1 to 0.05, $\sigma(t)$ decreasing from 3 to 0.7), the SOM structure becomes noticeably smoother. Figure 4.7 and Figure 4.8 depict the results for the cylinder-specific SOM used for Figure 4.3 and Figure 4.4 after performing a second training phase with the new grasp postures. The inter-node distance structure depicted in Figure 4.7(a) is very homogeneous. The second training phase resulted in more nodes that are unsupported by the original training data (see Figure 4.7(b)) but better represent the grasp postures generated by the grasping algorithm (see Figure 4.7(c)).

By comparing Figure 4.4(b,left) and Figure 4.7(b), it becomes clear that after the second training phase a noticeably smaller amount of nodes is used to represent the original training data resulting in a coarser posture resolution in these regions. On the other hand, since in

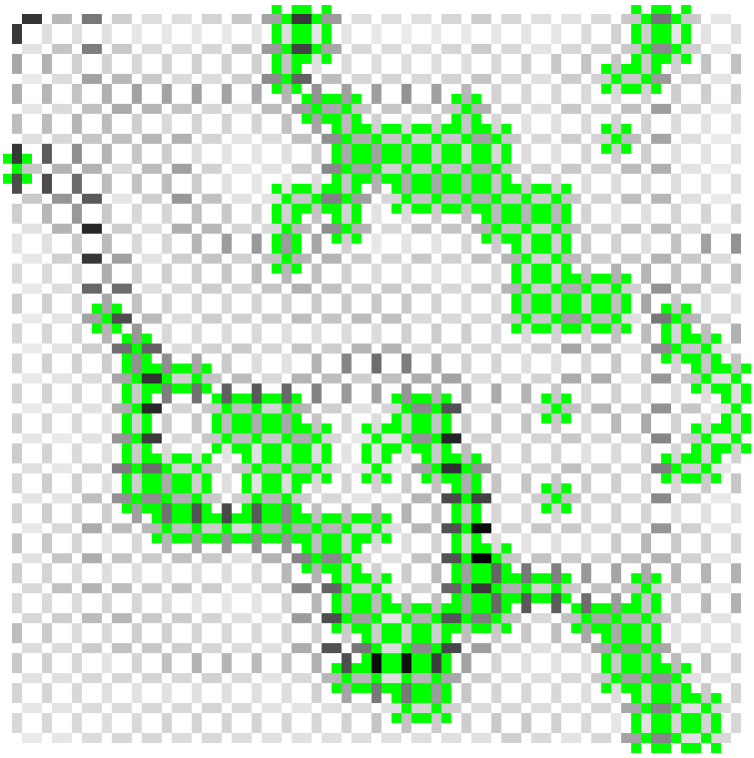
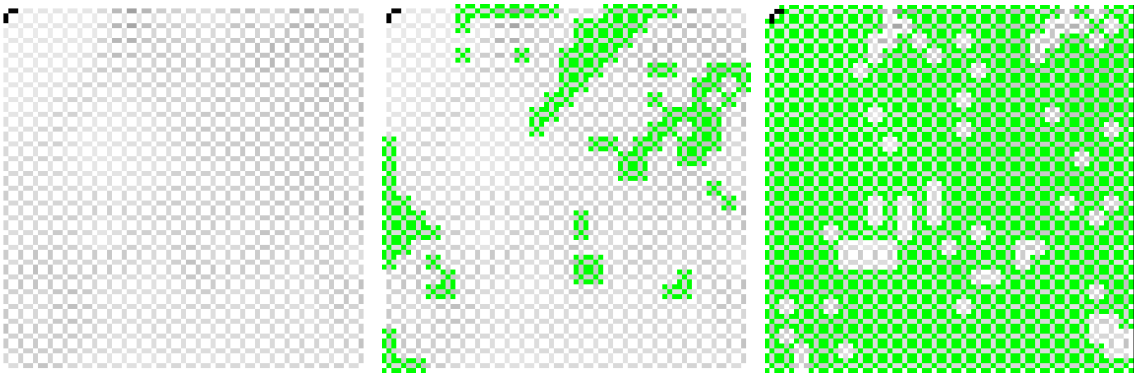


**Figure 4.6.:** Visualisation of the inter-node distances and the data-projection-nodes of the 25x25 cylinder-SOM. Depicted is Figure 4.4(a) extended by the node matches (green background) when projecting all successful grasp postures from evaluation (force closure grasps characterised by non-zero positive magnitudes of the worst-case disturbance wrench within the L1 grasp wrench space, see e.g. (Haschke et al., 2005)). The grasp postures match mainly on the inter-cluster nodes.



(a) (b) (c)

**Figure 4.7.:** Visualisations of the inter-node distances and the data-projection-nodes of the 25x25 cylinder-SOM used for Figure 4.3 and Figure 4.4 after resuming the learning with grasp postures generated by the experience-based grasp control. **(a)** besides one outlier (which was an outlier already before), the inter-node distance structure of the whole SOM is very homogeneous. **(b)** overlaid node matches (green background) when projecting the training data onto the SOM and **(c)** with overlaid node matches (green background) when projecting the new grasp postures generated by the control algorithm.

**Figure 4.8.:** 10x10 extract of the cylinder-specific 25x25 SOM *Grasp Manifold* used for Figure 4.3 and Figure 4.4 after resuming the learning with grasp postures generated by the experience-based grasp control. The node structure is now very smooth, the hand postures fade smoothly from one node to the next.

the testing phase a regular position/orientation grid was used covering the main part of the expedient position/orientation space, it is more desirable to represent the resulting test data with a stable resolution (as depicted in Figure 4.7(c)) than the clustered original training data in a very high resolution, under-representing the meaningful inter-cluster grasps that are not covered by it.

## 4.3. Grasp evaluation of the SOM Grasp Manifold

The evaluation of the grasp strategy including the execution of the grasps and the evaluation of their stability has been performed in a physics-based 3D computer simulation using the

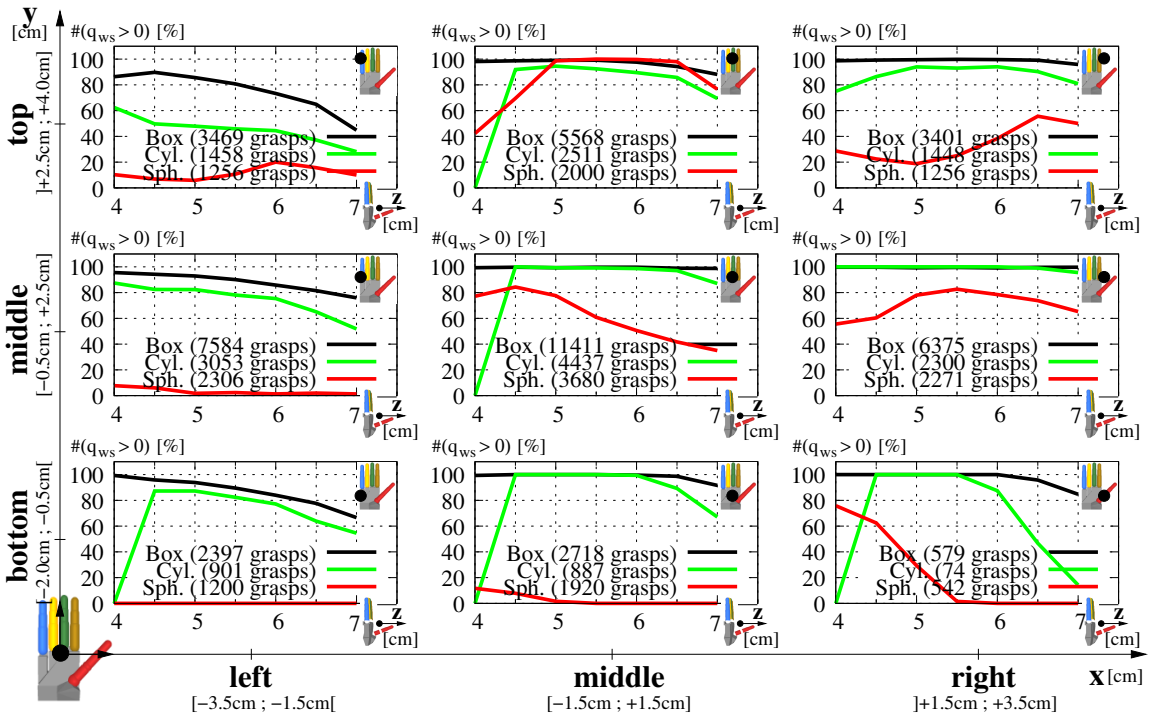



**Figure 4.9.:** Generalisation results for 77302 grasps (box: 43802, cylinder: 17069, sphere: 16431) in nine different object centre position regions in $x$-$y$-plane (parallel to palm; region see little figures in the upper right corners of each subgraph; position intervals: see overall-frame). Depicted are the percentages of accomplished force closure grasps ($q_{ws} > 0$) mapped onto the distance between palm and object centre (in z direction; z=4cm represents an object/hand distance near zero due to object and hand expansion). The number of grasps in each region is denoted in the according subgraph.

physics-engine Vortex (CM-Labs, 2005) and the graphical simulation toolkit Neo/NST (Ritter, 1990 – 2010). As grasp stability measure, the magnitude $q_{ws}$ of the worst-case disturbance wrench within the L1 grasp wrench space , see e.g. (Haschke et al., 2005), has been used. In particular, a value $q_{ws} > 0$ characterises a force closure grasp. The evaluation set-up consists of the *Shadow Dextrous Hand*-model (details see Section 3.1.1) and the object which is hovering in front of the hand due to zero-gravity conditions in the simulation environment (see Figure 4.1). While the palm is fixated in the world, the object can move freely.

To verify the generalisation ability of the discrete *Grasp Manifolds* with respect to position and orientation of the object, the evaluation of the grasp stability is conducted starting from various initial positions. These are selected from a regular grid spaced at 0.5 cm for cylinder and box, adding to a total of $15 \times 13 \times 7 = 1365$ positions, and spaced at 0.2 cm for the sphere resulting in $36 \times 31 \times 16 = 17.856$ positions. The grid dimensions can be read from the axis labels in Figure 4.9 and are illustrated in Figure 4.10. At each position the objects are also presented at $2 \times 4 \times 6 = 48$ different orientations (x: 0°, 45°; y: ±5°, ±15°; z: −45°– +5° spaced at 10°), where rotations about the x-axis are ignored for the cylinder and entirely for the sphere due to their symmetries. Initial configurations that already cause collisions are ignored, which results in a total of 43.802 box grasps, 17.069 cylinder grasps and 16.431 sphere grasps.

To visualise the grasping results within this high-dimensional test space as concisely as

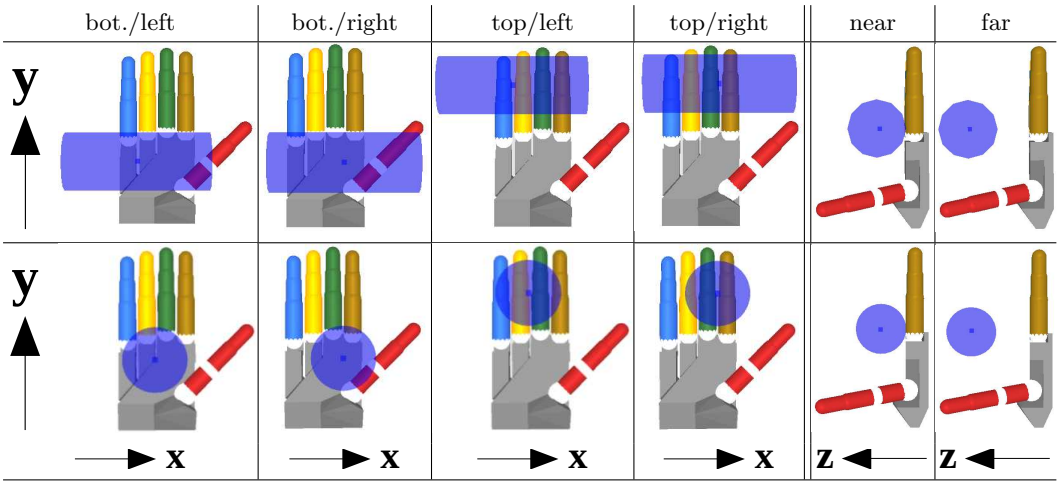



**Figure 4.10.:** Extremal initial positions of box/cylinder (first row) and sphere (second row) relative to the palm coordinate frame.

possible, Figure 4.9 displays the relative amount of force closure grasps (characterised by a grasp stability value $q_{ws} > 0$) averaged over all orientations and within nine subregions of the x-y-plane which is parallel to the palm plane. The distance of the object centre to the palm (along z-axis) is shown most detailed within the sub figures, because it strongest affects the grasping result. Altogether, 79.53% of all grasps are force closure (box: 93.93%, cylinder: 83.41%, sphere: 37.12%), i.e. are stable with respect to any (small magnitude) disturbance wrench.

Due to their similar object geometry, the evaluation outcome of the box and cylinder grasps are very similar in most object position regions whereas the number of force closure grasps for the cylinder stays almost always slightly below that for the box. Particular differences are on the one hand the lack of $z = 4$ values of the cylinder graphs in the middle and bottom positions caused by initial intersections with the hand at a centre distance $z = 4$ which are completely ignored for evaluation. On the other hand, a significantly faster decrease of the number of cylinder force closure grasps with increasing object/palm distance can be detected which is caused by a combination of a contact simulation stability problem and the occurrence of a net object force pushing the object out of the grasp in the case of large object/hand distances (see Figure 4.11). Additionally, the object can slide out of reach in the case of a misplaced first contact since the object movements are not constrained (e.g. by a table).

The sphere can be grasped only in a very restricted position subspace. On the left side, the most crucial reason for the loss of performance is that the required hand postures cannot be actuated appropriately (especially due to the restricted movability of the thumb) neither in the experience acquisition phase nor in the grasp execution itself. In the case of the right side regions, one drawback of the algorithm becomes apparent: since the algorithm cannot "see" the object before the first contact and it is not able to restart a grasp with another initial hand posture, the actuation of the situation-appropriate posture out of a disadvantageous initial posture can push the object apart and out of range. Similar to the box and the cylinder, the rate of force closure grasps decreases with increasing hand-object distance – again caused by a net contact force pushing the sphere away from the hand (Figure 4.11).

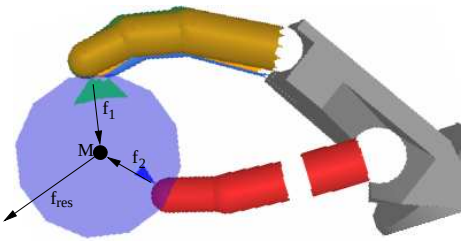



**Figure 4.11.:** If the fingers hardly reach round objects in distant positions, the net contact force $\mathbf{f}_{res}$ pushes the object away from the hand.

In comparison to the same evaluation scenario without the inclusion of the experience base, the "experienced" version achieves a performance gain of +14.85% of all grasps (box: +18.11%, cyl.: +5.10%, sph.: +14.35%). The most distinct performance differences can be observed in the non-centre regions in which the experience-based algorithm proves to be more robust concerning the object position or the pregrasp respectively.

The same evaluation scheme using one *object-unspecific* 30x30 SOM (trained with all data of all objects) for all grasps results in less force closure grasps, outperforming the purely tactile-driven algorithm only in the case of the box. Altogether, the object-specific SOMs achieve +7.87% force closure grasps (box: +5.73%, cyl.: +4.98%, sph.: +16.60%).

## 4.4. From grasping to manipulation

Grasping and manipulation are closely related. Nevertheless, the aims usually are quite different: in the case of grasping, the task is to fixate the object such that it cannot move within the grasping hand, while when thinking of manipulation, the aim is indeed rather to perform a certain movement with it.

In general, the task of object manipulation is more complex and diverse than the mere fixation and consequently, the fundamental ideas of the approaches also become more complex and diverse. Some examples for such approaches include the following:

Michelman and Allen (1994) implemented simple object translations and rotations with the Utah/MIT Hand and combined them to more complex tasks. In this manner, they achieved to remove a child-proof bottle top with two fingers exploiting a decomposition into subtasks and explicit force and position control schemes.

Zhang et al. (1996) define a graph of vertices representing *canonical grasps* consisting of topological hand/object feature pairs having contact when the associated grasp is achieved. Directed edges between two grasps represent possible transitions which have to be designed as lower-level control laws. Manipulation planning then is implemented as path planning in the graph between defined start and end vertices.

Fuentes and Nelson (1998) learn a mapping from *perceptual goals* – consisting of targeted object position/orientation and applied finger forces – onto robot commands realising these goals using an evolution strategy. Afterwards, manipulation can be performed by defining the task-specific perceptual goal and applying the learned mapping.

Han et al. (2000) propose a pure contact wrench analysis approach. They use a planner to generate a path in the space of *feasible configurations of the manipulation system* respecting hand/object constraints. A controller then incorporates sensor readings and system kinematics and statics to properly actuate the planned path.

Platt et al. (2004) address dextrous manipulation by sequencing concurrent combinations of hierarchically organised closed-loop controllers each derived from potential functions and realising force-related objectives. By operating subordinated controllers in the nullspace of superiors, higher-level conditions like wrench closure can be prioritised and thus sustained.

To a certain extent, all these approaches require the manual design of lower-level problem-specific controllers from scratch which is in general not desirable.

In this context, Schaal (1997) argues that learning without incorporating prior knowledge is a mostly artificial approach rarely taken by humans. He analyses the benefit of learning from demonstration and applies reinforcement learning on balancing a pole with an anthropomorphic robot arm to find an optimal policy. This problem can be solved based on data from a 30 second demonstration. Nevertheless, he concludes that not every learning problem can profit from prior knowledge in the same the way.

Pollard and Hodgins (2003) presented a different approach to incorporating human demonstration. They adapt quasi-static manipulation tasks to new friction conditions and untrained objects of known geometry by realising contacts and contact trajectories similar to former demonstration. The manipulation is planned such that the manipulator/object contacts combined with the extreme ground friction cones always produce force closure grasps arguing that intermediate configurations then are force closure too.

Although these approaches all realise robust dextrous manipulations to a certain degree, their implementations require considerable effort in problem modelling on the level of task definition and object characteristics.

## 4.5. Discussion

In terms of the described manifold representation, the statement from the last section – i.e. that *grasping* corresponds to object fixation whereas *manipulation* rather describes an object movement – can be reformulated as follows:

Grasping corresponds to "pulling" the current hand posture onto the manifold such that the resulting posture is *one point* on it which causes a stable immobilisation of the object. Hence, the goal is one specific final hand posture.

Manipulation in contrast is not the search for only one specific *point on* the manifold, but rather for a whole *trajectory through* an adequate manifold. Thus, the goal is a sequence of several intermediate hand postures which result in the targeted manipulation when actuated sequentially. In the initial phase, the hand posture needs to be pulled onto the manifold similar to the grasping process, but the resulting posture causes not necessarily an object immobilisation or not even object contacts.

The *Grasp Manifold* however is a good starting point for the construction of the *Manipulation Manifolds*. The conditions which have to be fulfilled indeed need to be adapted to the new task. The basic ideas of the *Grasp Manifold* realised in (Steffen et al., 2007a) are:

(a) every point on the manifold is a *grasp posture* being a hand posture that realises a grasp in combination with the corresponding object in the corresponding position.

(b) the manifold spans over the whole targeted workspace or over all targeted grasping contexts, respectively. The grasp algorithm would not be able to perform well in all targeted contexts otherwise.

(c) the manifold representation needs to support the *associative completion* mechanism (Steffen et al., 2007a; Walter, 1996) which enables the projection of partially specified

data points onto the manifold resulting in the completed hand postures lying on the manifold. This allows for a dynamic incorporation of the tactile information.

(d) the sample resolution in the represented manifold realised by prototypes or reference vectors need to be high enough. Nevertheless, a continuous manifold representation is not necessary because a generic closing algorithm is able to terminate the grasp starting from a hand posture near a final grasp posture.

In the case of *Manipulation Manifolds*, these conditions have to be modified in order to fit to the new task of representing motions or sequences of intermediate hand postures instead of only single points:

(a) To represent the pure manipulation phases – thus only those parts of the manipulation in which the hand is in direct interaction through contacts with the object – it is again necessary that every point corresponds to a contact situation. However, in the Manipulation Manifold approach, the aim is to represent whole manipulation movements possibly containing non-contact phases. Hence, the previous requirement can be weaken to: the resulting manifold consists only of points corresponding to hand postures generated by the manipulation movement to be represented. The *grasp* postures demanded by the *Grasp Manifold* conditions are comprised herein. Additionally, only hand postures representing contact situations (not necessarily grasp postures) implicitly inhere object information and thus, only these postures can be used later to infer object-specific parametrisation of the manipulation movement.

(b) For manipulation, it is still necessary to represent the whole subspace of the targeted manipulation movement to enable an algorithm to reproduce it.

(c) In the grasping case, the associative completion is used to "pull" the current hand posture onto the manifold. Basically, this is not required in order to perform the manipulation movement by navigating through the manifold. Nevertheless, to find a good starting point, it is a good approach to perform an initial "pulling" onto the manifold similar to grasping. This, however, necessitates again the associative completion mechanism.

(d) Since the target is not only one specific point but a navigation through the manifold following a trajectory where every intermediate hand posture lies on the manifold, a discrete approximation like the SOM lacks the necessary precision. Instead, a continuous manifold representation is used.

If condition (c) is guaranteed and the regions of the manifold described by hand postures in the training data that effect object contacts are specially marked – the *Grasp Manifold* is a subset of the *Manipulation Manifold* and grasping and the reproduction of manipulation movements can be combined in one representation. In this combination, the initial grasping can be seen as object-specific parametrisation of the subsequent manipulation movement.

# 5. Structured Manifolds for Motion Production

The vision of the robot helper that assists the human in his every-day life has been the motor for research in many domains and since many years. One of the key challenges on the way to making this vision real is the possibility to transfer human skills to dextrous robots in an easy, fast and robust manner.

In pursuit of this far reaching goal, a lot of work has been carried out in robotics in the fields of imitation learning and learning from observation/demonstration (Atkeson & Schaal, 1995; Bentivegna et al., 2004; Breazeal & Scassellati, 2002; Demiris & Hayes, 2002; Ijspeert et al., 2003; Schaal, 1999; Schaal et al., 2003).

Perhaps the simplest form of transferring skills to a robot consists of directly copying the motor commands of the demonstrator to the robot. Whereas this approach can be very effective, its application generally turns out to be impossible in most cases since the motor commands are either not available or inappropriate for the robot.

While extensive research effort in imitation learning is indeed focussed on overcoming these two major problems, the approach presented in this section follows the idea of Oztop et al. (2006), who studied the skill transfer from human instructors to a robotic platform. They proposed to consider the robot as a tool for the human such that the human includes the robot in his body schema. In consequence, he automatically uses his visuo-motor learning capacity for generating appropriate training data directly on the robot.

Following this paradigm, Oztop et al. (2006) recorded data from a robot hand - controlled by a human via a motion capture system that performs the swapping of Chinese health balls. Afterwards, they represented this manipulation using an open-loop controller that achieves the same ball swapping task without human guidance. The method to represent the manipulation was however highly optimised for this one specific application and cannot be transfered to a simple and robust approach of general nature.

The remainder of this chapter presents an approach to representing data that has been recorded following the same paradigm, but in a more general, and also highly structured manner. To this end, manifolds of Unsupervised Kernel Regression (see Section 2.1) are learned in a *partly* supervised manner yielding a manifolds representation of the manipulation whose inherent structure can be exploited for the later motion generation.

Before going into the details of the representation characteristics and the possibilities to create them, the manipulation data that has been used to illustrate the different aspects of the approach is presented in the next section. At the end of this chapter, the loop back to the same ball swapping task that has been studied by Oztop et al. (2006) will present a sophisticated control scheme for the ball manipulation on the basis of the new approach.

A brief overview of related work and general considerations about required characteristics of a manifold representation of manipulation motions can be found in the previous chapter. There, the idea of such manifolds is motivated on the basis of Grasp Manifolds, and in particular, the development of the idea of Grasp Manifolds towards the idea of Manipulation Manifolds is described in detail.

**Figure 5.1.:** Visualisation of an exemplary sequence of a recorded cap turning manipulation movement. The illustrated hand postures are generated in a physics-based simulation, controlled via a data glove. Joint angle corrections are incorporated from the collision detection module of the simulation software which avoids a penetration of the fingers and the bottle cap, that was present in the recording scene.

The work presented in this chapter is partly based on already published material (Steffen et al., 2008b, 2008a; Steffen, Klanke, et al., 2009a, 2009b).

## 5.1. Manipulation Data

One usually critical issue of methods for manifold learning or non-linear dimensionality reduction, respectively, is that the task-related performance strongly depends on the underlying training data. Although UKR is able to handle noise in the training data to some extent, the data collection should nevertheless be conducted thoroughly to improve learning results.

In the case of dextrous grasping and manipulation with a robot hand, one of the main problems is to generate adequate training trajectories of hand postures for a new, previously unknown manipulation movement. Per-joint control or directly physically moving the fingers by hand does not yield natural trajectories since it is usually not possible to coordinate up to 24 variables at once in an adequate manner. A viable alternative that yields good data is to use human hand data from a motion capture system.

Concerning the recording of robot-appropriate data, the approach described in this chapter follows the idea of Oztop et al. (2006, 2007). They propose to consider the robot as a tool for the human and to exploit the human learning capacity to use this tool for generating appropriate training data directly on the robot. For the experiments presented later on, hand posture data has been recorded with a data glove – an Immersion CyberGlove II with 22 bend sensors for the different joints (see Section 3.6). The first step is to map the sensor values onto a simulated hand model and to perform the data generation in a simulated manipulation scenario. On the one hand, joint angle corrections are incorporated which are provided by the collision detection of a physics-based simulation toolkit CMLabs Vortex[1]. On the other hand, more general hand posture corrections are provided by the user induced by visual feedback.

With this indirect method, sequences of hand postures during cap turning movements for five different cap radii ($r = 1.5cm,\ 2.0cm,\ 2.5cm,\ 3.0cm$ and $3.5cm$) have been recorded. For each of these radii, five to nine sequences of about 30 to 45 hand postures each have been recorded – in total 1204 for all sequences and all radii. Each hand posture consists of a vector of the 24 joint angles of the simulated hand model.

An exemplary sequence is illustrated in Figure 5.1.

[1] http://www.vxsim.com

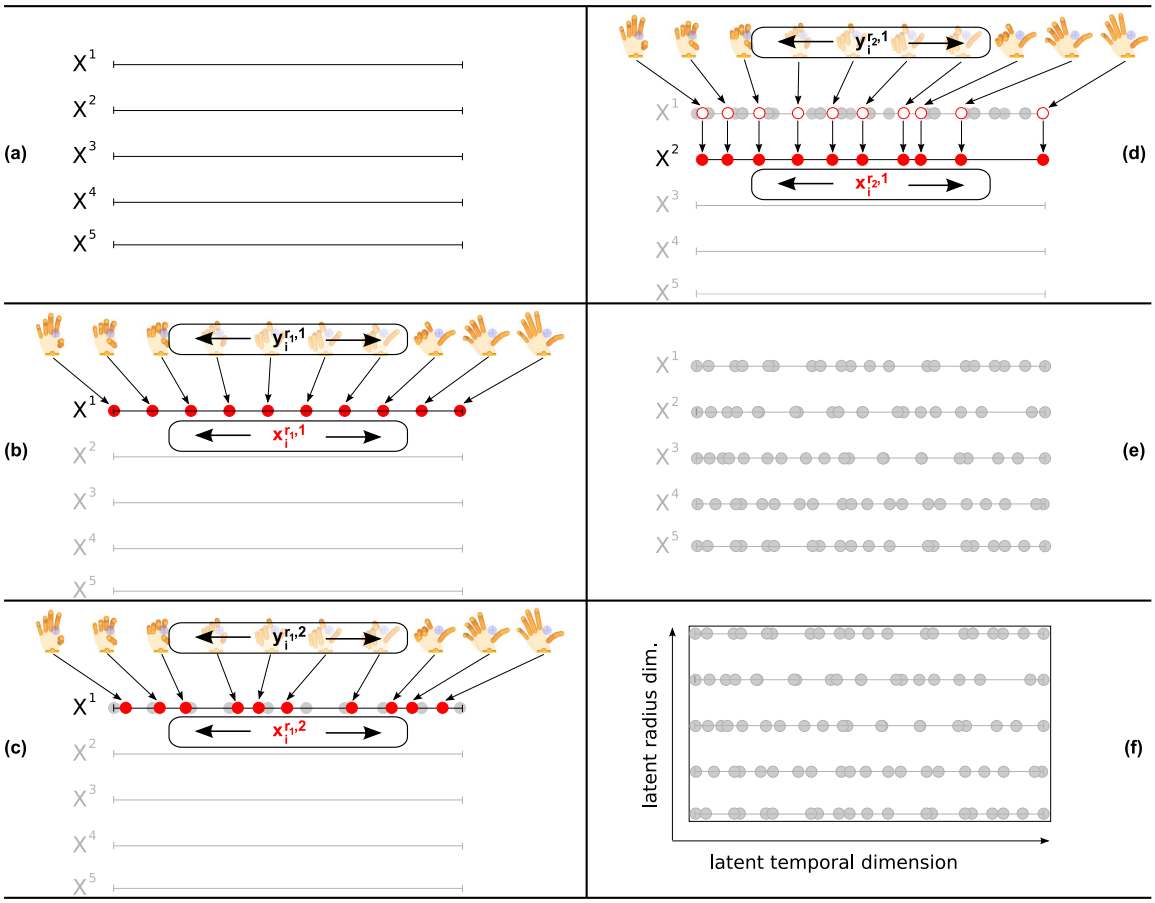



**Figure 5.2.:** Schematic description of different steps in the manifold construction process.

## 5.2. Manifold Generation by Construction

In the example of turning a bottle cap, which will serve as prototype manipulation in the following sections, the goal is to realise a manifold in which one latent dimension controls the motion progress in time and another dimension specifies the radius of the cap. Performing the movement then reduces to modifying the time component of the latent parameter while keeping the radius value fixed.

A simple but effective approach is to construct the final manifold out of several sub-manifolds each realising a manipulation movement of one specific motion parameter set. In the example of turning a bottle cap, such parameter set e.g. consists of the cap radius of the underlying recorded manipulation and thus, the progress in time of the movement and the radius constitute the targeted latent dimensions.

The construction of the final manifold is performed iteratively. It starts with training sequences corresponding to the minimal cap radius and successively increases the radius of the subsequent sequences. The iterative construction is performed as follows:

1. Initially, one empty one-dimensional latent space for each recorded radius is provided (see Figure 5.2(a)) which are to be filled incrementally with data corresponding to the recorded sequences.

2. The first sequence of hand postures $\mathbf{Y}^{r_1,1} = \{\mathbf{y}_i^{r_1,1}\}$ corresponds to the minimal cap

radius $r_1$. The associated latent parameters $\mathbf{X}^{r_1,1} = \{x_i^{r_1,1}\}$ of a 1D-UKR manifold are initially equidistantly distributed in a predefined interval according to the chronological order of the hand postures (see Figure 5.2(b)). The resulting UKR manifold is denoted as $\mathcal{M}^{r_1,1}$.

3. The incorporation of the second sequence (representing another example of a movement for the same radius) is performed in an iterative manner: the hand posture vectors $\mathbf{Y}^{r_1,2}$ of this sequence are projected pointwise into the latent space of the previously trained 1D-manifold $\mathcal{M}^{r_1,1}$ resulting in $\mathbf{X}^{r_1,2}$ (see Figure 5.2(c)). Using this projection, a synchronisation of the temporal advance of the two movements is realised.

4. In the next step, the different sets of hand posture data as well as latent parameters are combined to one new UKR manifold $\mathcal{M}^{r_1,\{1,2\}}$ containing all observed data $\mathbf{Y}^{r_1,\{1,2\}} = \mathbf{Y}^{r_1,1} \cup \mathbf{Y}^{r_1,2}$ and the corresponding latent parameters $\mathbf{X}^{r_1,\{1,2\}} = \mathbf{X}^{r_1,1} \cup \mathbf{X}^{r_1,2}$. A subsequent UKR training of $\mathcal{M}^{r_1,\{1,2\}}$ optimises the latent parameters subject to the whole combined data set.

By performing this procedure for all sequences of hand postures corresponding to cap turning movements for one specific cap radius, a 1D–UKR is trained representing a generalised radius-specific movement. The application of this method to all sets of radius-specific sequences yields one 1D–UKR per training radius. In order to promote the synchronisation of the temporal advances also between the different radius-specific manifolds, only the first manifold $\mathcal{M}^{r_1,1}$ is initialised with equidistant latent parameters as described above. When proceeding with a sequence $\mathbf{Y}^{r_{(k+1)},1}$ of a new radius $r_{(k+1)}$, it is projected onto the previously trained manifold $\mathcal{M}^{r_k,\{1,..,n\}}$ (as with sequences for the same radius). The resulting latent parameters are utilised as initialisation $\mathbf{X}^{r_{(k+1)},1}$ of the new manifold $\mathcal{M}^{r_{(k+1)},1}$ instead of being combined to a manifold $\mathcal{M}^{r_k,\{1,..,n+1\}}$ (see Figure 5.2(d)). The training continues as described above.

In consequence, each first sequence used to initially train a new 1D-manifold plays a special role and determines the relevant subspace in the hand posture space. Therefore, it is important that these first sequences represent complete movements.

The subsequent combination of all final 1D-manifolds $\mathcal{M}^{r_i,\cdot}$ (Figure 5.2(e)) to one 2D-manifold $\mathcal{M}$ which represents the complete movements for all training radii $r_i$ is performed manually and without the usage of the UKR optimisation.

Finally, $\mathcal{M}$ consists of all incorporated training data $\{\mathbf{Y}^{r_i,j}\}_{i,j}$ together with the corresponding latent parameters $\{\mathbf{X}^{r_i,j}\}_{i,j}$ and represents the whole manipulation movement covered by the training data and therefore constitutes the *Manipulation Manifold*. The extension to two dimensions is realised by expanding each one-dimensional latent parameter $x_i$ to a two-dimensional vector $\mathbf{x}_i = (x_i, r_i)^T$ incorporating the appropriate radius corresponding to the associated training sequence (Figure 5.2(f)).

### 5.2.1. Construction results

The method described in the previous section has been applied to all recorded training sequences starting with the sequences corresponding to the minimal radius and successively incorporating sequences of larger radius values. After having trained one 1D-manifold for each of the training radii in the described synchronised manner, the corresponding radius

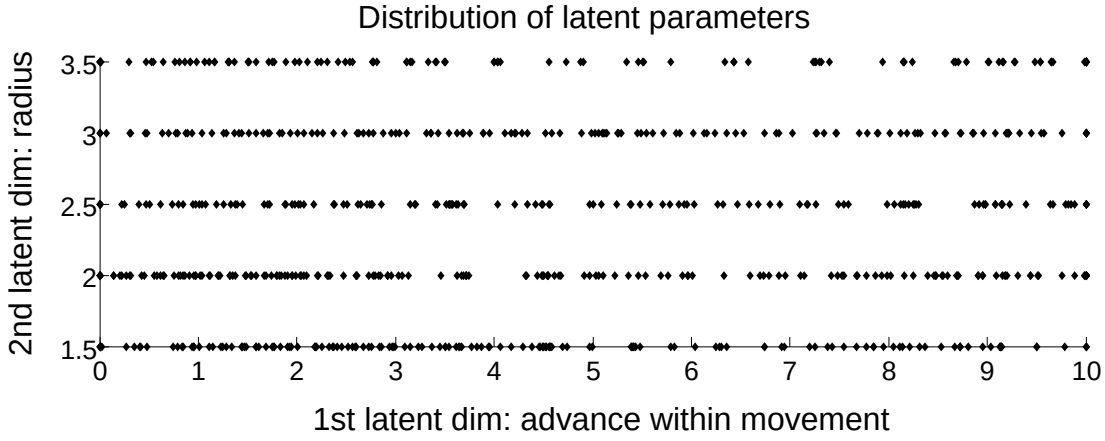



**Figure 5.3.:** Distribution of the latent parameters of $\mathcal{M}$. 1*st* dim: temporal advance within the movement. 2*nd* dim: cap radius.

values have been added as second dimension to the latent parameters (by manually extending each latent parameter by an extra dimension). The distribution of the latent parameters in the new latent space is depicted in Figure 5.3. As constructed, the horizontal dimension represents the temporal advance within the cap turning movement and the vertical dimension denotes the associated cap radius. Since no further UKR training is performed, the latent parameters only lie on the previously set discrete radius values.

The evaluation of the approach is based on the analysis of the qualitative ability to represent the underlying manipulation motion. Since a reproduction of the represented manipulation will require an adequate controller that also takes the contact situation into account, the manifold basis rather constitutes a guideline of how the fingers need to be coordinated. The very fine aspects depend on very subtle details of the object, the object contacts, and the object-hand relation, which cannot be recorded by means of the available devices and therefore not represented either. An evaluation on the basis of reconstruction errors is therefore of minor interest for the manipulation task.

To get a more intuitive impression of the movements represented by the manifold and its generalisation abilities, Figure 5.4 depicts a matrix of hand postures corresponding to the latent positions in a regular grid covering the latent space of the manifold. Again, the temporal advance is depicted in the horizontal and the different radii in the vertical direction. To facilitate the comparison, a bottle cap with radius $r = 1.5cm$ is depicted in each sub-figure. As shown in Figure 5.3, only the radii $r = 1.5cm,\ 2.0cm,\ 2.5cm,\ 3.0cm$ and $3.5cm$ are directly supported by training data. The depicted intermediate radii in Figure 5.4 therefore visualise the generalisation ability of the constructed manifold to new cap radii.

The corresponding movements for the intermediate radii are of similar nature as the recorded training sequences. They only differ in the degree of hand closure in the intermediate phase and therefore in the associated cap radius.

The figure also illustrates the effect of the temporal synchronisation between the different 1D-manifolds by projecting new sequences into the latent space of the previously trained manifolds. The most distinct picture of this synchronisation can be seen in columns $3-5$ of Figure 5.4 where the fingers are shown in the time steps when they contacted the cap

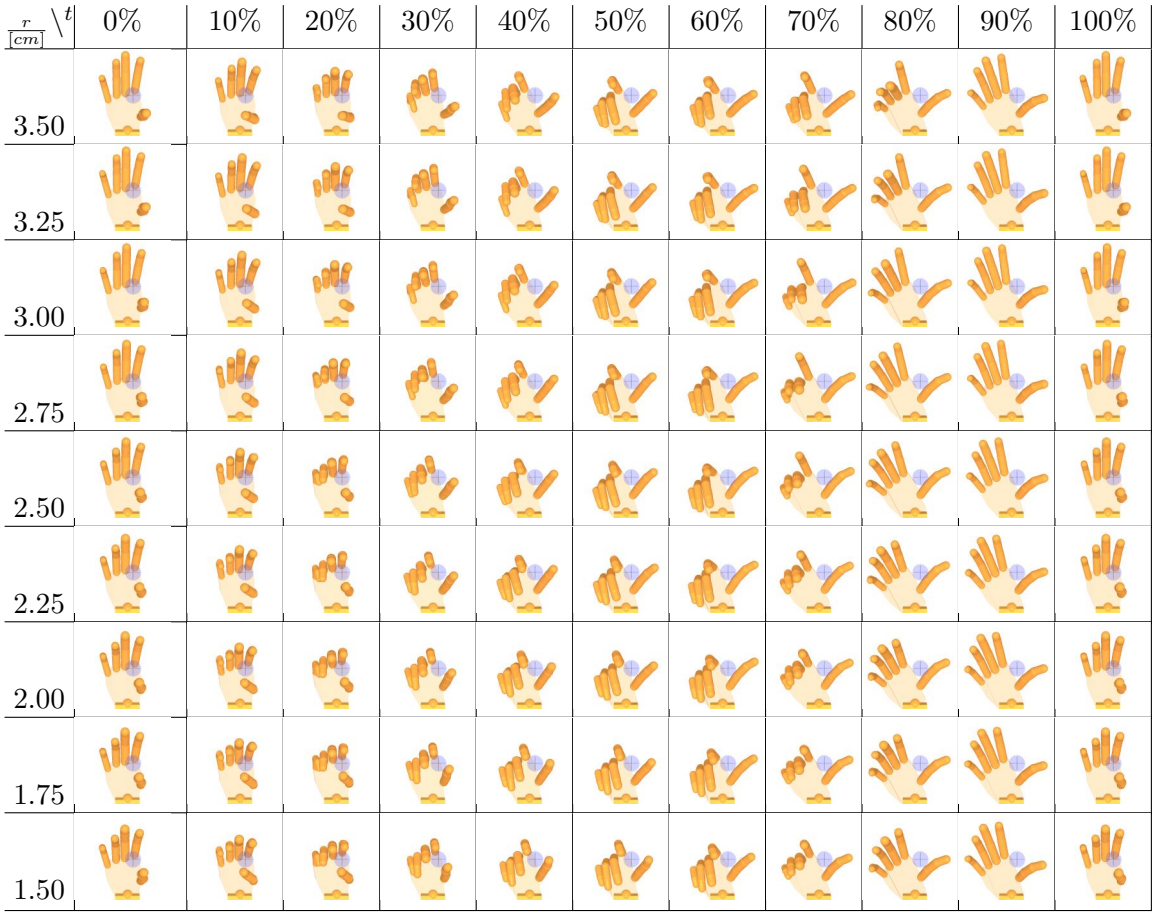



**Figure 5.4.:** Generalisation results of the UKR training/construction. The hand postures correspond to positions of a regular grid covering the latent space. The row direction correspond to the temporal dimension; the column direction to the radius dimension. The bottle caps ($r = 1.5$) are depicted only as a comparison aid. The radii $r = 1.5, 2, 2.5, 3, 3.5cm$ correspond to radii covered by training data, $r = 1.75, 2.25, 2.75, 3.25cm$ demonstrate the generalisation.

during the recording of the motion. Additionally, those columns give an impression of the smoothness in the $2nd$ manifold dimension. Finger openings are smooth with increasing cap radius in the column direction. In the row direction, all depicted examples provide smooth transitions from left to right indicating a smooth manifold also in the time direction.

The turning movement now can be easily performed repeatedly. The corresponding control schemes are described in detail in Section 5.4 to 5.6. For the constructed manifold, the repeated actuation of the represented motion requires a reset of the temporal latent position to the beginning when the manifold border has been reached in the temporal dimension (go back from 100% to 0%). Since there is no regulation for border synchronisation incorporated in the construction, the 100%- and 0%-postures usually significantly differ from each other yielding an abrupt non-smooth hand movement when jumping back to the 0%-posture. Since the beginning and the end of the movement are the phases where the fingers are the farthest away from the cap, this motion artifact does not effect the cap turning. It is therefore not of particular relevance for the success of this kind of manipulation. However, this issue has been accounted for by introducing periodic latent dimensions, as will be described in Section 5.3.

### 5.2.2. Discussion

The method to construct *Manipulation Manifolds* has originally been developed for the purpose of proving the concept of a structured manifold representation for motions. The algorithm is therefore not perfectly "clean" in some points like the construction of latent parameters by manually adding dimensions. In addition, issues like the latent interval in the time dimension or the bandwidth in the radius dimension have to be handled carefully.

Nevertheless, the resulting manifold constitutes a very useful motion representation which is easy to generate from sequences of training data, which can easily incorporate prior knowledge about the underlying task and which provides easy means to recover the represented motion for a later actuation e.g. on a robot.

For the case that less specific prior knowledge is provided beforehand, the next section will address the automatic *learning* instead of construction of the presented structured manifolds. To this end, extensions and modifications to UKR learning are discussed and evaluated.

## 5.3. Manifold Generation by Learning[2]

Learning of control manifolds is emerging as one of the key challenges in unsupervised learning. The Self-organising Map (SOM) has been influential in various pertinent approaches (see e.g.(Barreto et al., 2003)). Unsupervised Kernel Regression can be seen as a successor bridging between earlier "Parametrised SOM" (PSOM, (Walter & Ritter, 1996)) and kernel methods (e.g.(Schölkopf et al., 1998)).

In the previous section, UKR was shown to be well suited for representing human manipulation data. However, due to UKR being unable to incorporate prior knowledge about the data structure in an automatic manner[3], generating *Manipulation Manifolds* from training sequences of hand posture data has been realised as supervised *construction* instead of automatic *learning*. In this section, extensions to UKR are described which enable the method to learn manifold representations of (periodic) sequences of chronologically ordered data. In order to realise the same kind of structured manifold that was described in the previous section, the new extensions need to automatically regularise the UKR objective function such that the trained model reflects the sequence structure of the data. By adequately extending the objective function (2.12) or (2.14), respectively, this regularisation realises the targeted *learning* of the Manipulation Manifolds.

As basis for several error measures and thus a systematic evaluation of the new extensions, an initial analysis on appropriate toy data which mimic the intrinsic characteristics of the targeted manipulation data has been performed. Whereas toy data always bare the risk of lacking transferability to the real data case, the thorough construction of similar data characteristics yielded results that could be transferred to the domain of dextrous manipulation, as will be shown later on.

The following sections are based on previously published material, i.e. (Steffen, Klanke, et al., 2009a, 2009b).

[2] The work presented in this section is partly based on the work developed by the author in collaboration with Dr. Stefan Klanke during a research stay in 2008 at the School of Informatics, University of Edinburgh, UK, in Prof. Dr. Sethu Vijayakumar's group.

[3] Indeed, Klanke and Ritter (2007) introduced the possibility to utilise $\varepsilon$-insensitive loss functions. This also provides the possibility incorporate prior knowledge about the data structure, but in a fundamentally different way as it is required here.

### 5.3.1. Extension of UKR for Representing Sequences of Data

To enable the originally purely unsupervised UKR training to benefit from prior knowledge about the structure of the observed data, extensions have been introduced which (a) especially consider ordered data *sequences*, (b) explicitly allow for *periodic* sequences, (c) propagate the original chronological intra-sequence order to their latent representations and (d) propagate stability of non-temporal sequence parameters within the sequences.

(a) Previously known affiliations of data points to sequences are considered. This enables to influence the latent parameter adaptation such that sequence-specific mechanisms can be involved in the training. To this end, one latent (temporal) intra-sequence dimension is explicitly distinguished from other inter-sequence (sequence parameter) dimensions (e.g. the cap radius in the bottle cap example).

(b) Periodic sequences consist of one periodic temporal and one/several (usually) non-periodic dimensions. To allow for such structure, different univariate kernels $k_l$ for different latent dimensions $l$ are provided. The basis functions $(\mathbf{B}(\mathbf{X}))_{ij}$ (2.13) then consist of their normalised products (parametrised by $\boldsymbol{\beta}_l$):

$$(\mathbf{B}(\mathbf{X}))_{ij} = \frac{\prod_{l=1}^{q} k_l(x_{i,l} - x_{j,l}; \boldsymbol{\beta}_l)}{\sum_k^N \prod_{l=1}^{q} k_l(x_{k,l} - x_{j,l}; \boldsymbol{\beta}_l)}. \tag{5.1}$$

In the non-periodic case, the univariate versions of the kernels used in original UKR can be applied, e.g. Gaussian:

$$k_g(x_i - x_j; \beta) = \exp\left[-\frac{1}{2}\beta^2(x_i - x_j)^2\right]). \tag{5.2}$$

In analogy to original UKR, no need for bandwidth control can be assumed. However, in order to analyse potential cross-effects with the following new extensions, the performance of different bandwidths is investigated in Section 5.3.3 as well for this case.

For the periodic case, the following cyclic $\sin^2$-kernel with bandwidth parameter $\beta$, periodic in $[0; \pi]$ can be used:

$$k_{\circlearrowleft}(x_i - x_j; \beta) = \exp\left[-\frac{1}{2}\beta^2 \sin^2(x_i - x_j)\right]. \tag{5.3}$$

Up to normalisation and scaling, the kernel is equivalent to the von Mises distribution (Mardia, 1972) which has for instance been used by Bishop and Legleye (1995) in order to represent periodic data characteristics. The adapted version in Equation (5.3) has been chosen for convenience reasons. Figure 5.5 depicts the $\sin^2$-kernel (5.3) for different choices of the bandwidth parameter $\beta$.

In the periodic case, kernel bandwidth regulation is needed since the effective space in corresponding dimensions is constrained due to its periodic nature and fixed bandwidths cannot be compensated for only by scaling the latent parameters.

(c) In order to propagate the original chronological order of $N_S$ data sequences $\mathcal{S}_\sigma = (\mathbf{y}_1^\sigma, .., \mathbf{y}_{N_\sigma}^\sigma),\ \sigma = 1..N_S$ to the corresponding latent parameters $(\mathbf{x}_1^\sigma, .., \mathbf{x}_{N_\sigma}^\sigma)$, the values $x_{i,d_t}^\sigma,\ i = 1..N_\sigma$ in the *temporal* latent dimension $d_t$ need to reflect the order of the original data sequence.

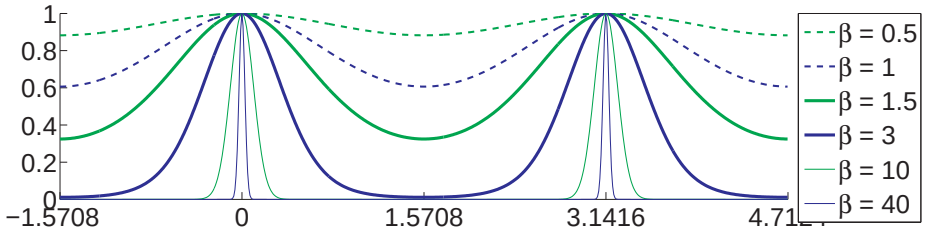



**Figure 5.5.:** The $\sin^2$-kernel for different choices of the bandwidth parameter $\beta$. The kernel is periodic in $[0; \pi]$.

In the periodic case, such condition is difficult to induce without any assumptions about the underlying sequences. However, by providing sequences of complete cycles, the first data point in the sequence can be considered as successor of the last one: $\mathbf{x}_0^\sigma = \mathbf{x}_{N_\sigma}^\sigma$. The following penalty term in the objective function then can preserve the cyclic data order:

$$E_{cseq}(\mathbf{X}) = \sum_{\sigma=1}^{N_\mathcal{S}} \sum_{i=1}^{N_\sigma} \sin^2(x_{i,d_t}^\sigma - x_{(i-1),d_t}^\sigma). \tag{5.4}$$

(d) One strong design choice in the construction of the *Manipulation Manifolds* as described in Section 5.2 is the clear structure in the latent space. Especially one latent dimension is designed to constitute the time (or phase) direction of the represented motion and the other latent dimensions are designed to represent distinct motion parameters. In the learning approach, these strong constraints can be propagated to the latent space by enforcing the values of the non-temporal dimensions to be approximately constant within single sequences. This consideration stems from the generation of the manipulation data. The basic idea is that the underlying movement parameters usually do not change during single sequences – e.g., for cap turning, the radius of the cap does not change during the turning.

Such regularisation of intra-sequence parameter variations can be realised again as a penalty term to the objective function which, in this case, penalises high variances in the non-temporal dimensions $k = 1..q,\ k \neq d_t$:

$$E_{pvar}(\mathbf{X}) = \sum_{\sigma=1}^{N_\mathcal{S}} \sum_{k \neq d_t} \frac{1}{N_\sigma} \sum_{i=1}^{N_\sigma} \left(x_{i,k}^\sigma - \langle x_{\cdot,k}^\sigma \rangle\right)^2 \tag{5.5}$$

The overall objective function can be denoted as

$$E(\mathbf{X}) = R(\mathbf{X}) + \lambda_{cseq} E_{cseq}(\mathbf{X}) + \lambda_{pvar} E_{pvar}(\mathbf{X}). \tag{5.6}$$

The additional parameters of the extended version of UKR are $(\beta_1, \ldots, \beta_q, \lambda_{cseq}, \lambda_{pvar})$.

### 5.3.2. Toy Data for Evaluation

To evaluate the new UKR extensions, a set of appropriate toy data with similar intrinsic characteristics as the manipulation data described in Section 5.1 has been generated. The utilisation of such toy data provides knowledge about the underlying true structures of the

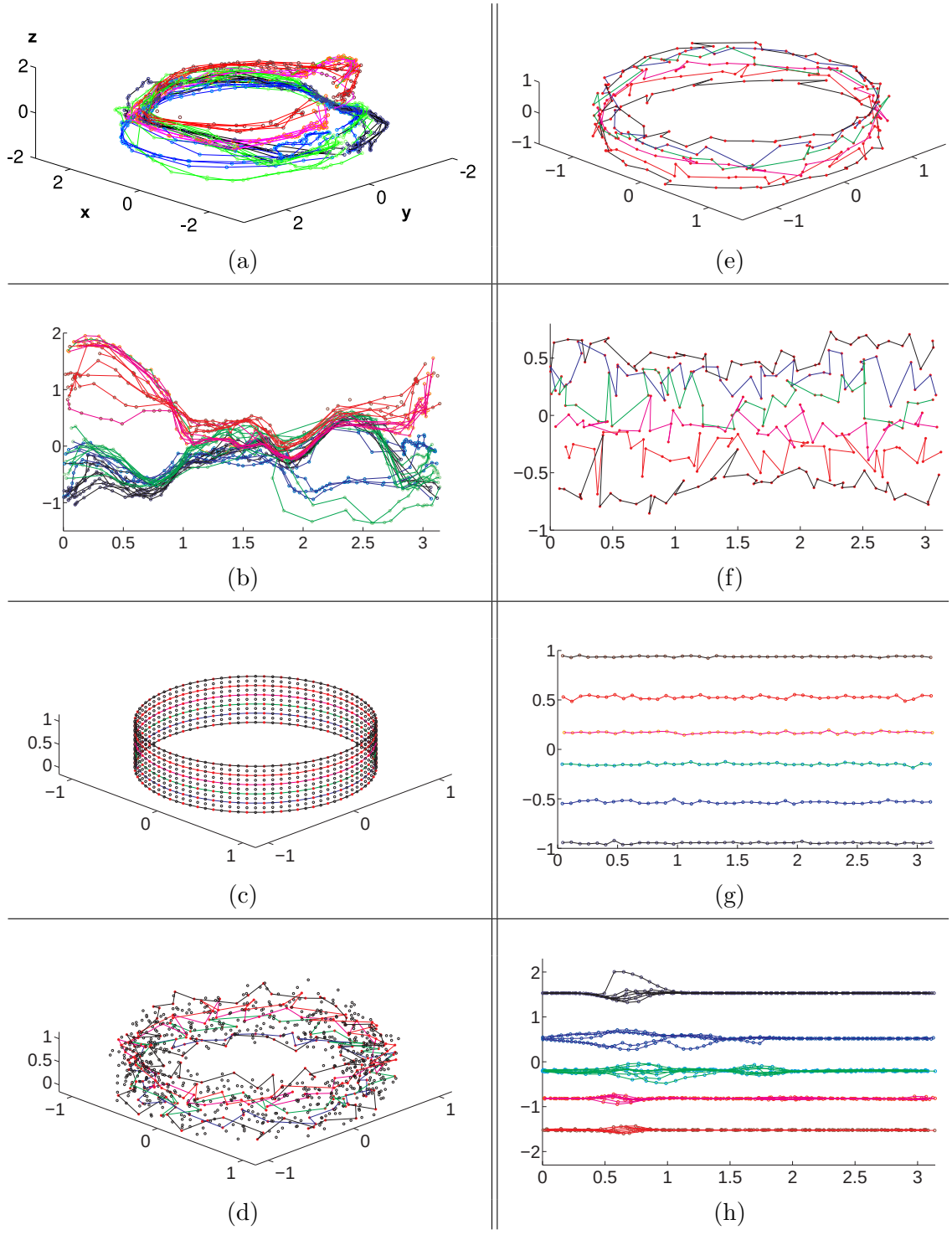



**Figure 5.6.:** **(a)** 3D Isomap embedding of 24D hand posture data recorded during the turning movement of a bottle cap. Different colours encode different cap radii. **(b)** atan2⋆-mapping of (a). **(c)** noise-free training data (red, connected); test data (black, single points). **(d)** noisy training/test data. **(e)** Toy data Isomap embedding (see (a)). **(f)** atan2⋆-mapping of (e). **(g-h)** Results for toy (g) and real (h) data.

evaluated data and therefore enables to compute a variety of error measures which could not be provided otherwise (see Section 5.3.3 for details).

As basis for an adequate toy data generation, the real manipulation data have been thoroughly investigated. Especially a focus has been put on uncovering the intrinsic data structures reflecting the prior knowledge of the generated manipulation data. From the recording of the real data, the existence of a periodic structure reflecting the periodic nature of the cap turning movement and an additional non-periodic expansion reflecting the different cap radii used in the recording can be expected. Indeed, by using the spectral embedding method Isomap ((Tenenbaum et al., 2000), see also Section 2.1.5) – a powerful method when it comes to uncovering hidden intrinsic structures in large data sets – the expected structure becomes apparent: a three-dimensional Isomap embedding of the manipulation data is depicted in Figure 5.6(a). It reveals a cylinder-like structure describing a periodicity living in the $x/y$ dimensions and a non-periodic extension in $z$ direction.

In order to unfold the 2D representation of the periodicity, e.g. atan2 can be applied on the $x/y$-part of the embedding data yielding the basis for the corresponding 1D angle. In order to fit to the periodic kernel $k_{\circlearrowleft}(\cdot)$ (5.3), a modified version of atan2 is used:

$$\text{atan2}^{\star}(\cdot) = \frac{\pi}{2} + \frac{1}{2}\,\text{atan2}(\cdot) \in [0; \pi]. \tag{5.7}$$

In combination with the original $z$ component, this yields a 2D representation of the formerly 3D Isomap embedding and of the 24D original hand posture data, respectively. This data can be used as latent initialisation of the UKR model[4] as visualised in Figure 5.6(b). It turns out that the different sequences (connected) are not clearly separated and even sequences corresponding to different cap radii (encoded by different colours) partly overlap.

In order to reflect similar characteristics in the toy data and to provide an informative basis for the subsequent evaluation, the aim is a simple low-dimensional toy data structure that produces Isomap embeddings of a similar form as the real data. To this end, ordered (connected) data samples from the surface of a cylinder geometry (height=1, radius=1, Figure 5.6(c)) living in 3D have been generated together with noisy versions (Gaussian noise, $\sigma = 0.1$, e.g. Figure 5.6(d)). Such data yield Isomap embeddings which

(a) provide a periodicity

(b) a non-periodic *parameter* expansion and

(c) are organised in *chronologically* ordered sequences ("trials")

and thus are quantitatively similar to the Isomap embedding of the real data (see Figure 5.6(a)/(e)) and its 2D mapping (see Figure 5.6(b)/(f)). Within this cylinder structure, cross sectional rings of different height levels model sequences for different cap radii in the real data. As basis for the evaluation, six training data rings and six overlapping together with five intermediate test data rings (see Figure 5.6(c)) have been generated.

In anticipation of the following, Figure 5.6(g-h) depict the resulting latent parameters from training with toy and real data, respectively, having considered the results from the toy data evaluation. The similarity of both latent structures supports the appropriateness of the toy data for the use with the real manipulation data later on.

[4]The 2D latent space with one periodic kernel has the topology of a cylinder surface.

### 5.3.3. Toy Data Evaluation and Results

After having generated appropriate toy data as described in the previous section, the data can be split into training and test sets. These serve as a basis for a systematic analysis of the characteristics and behaviours of the UKR extensions. The prior knowledge about the toy data, i.e. that they consist of periodic sequences with non-periodic height levels, corresponds to a periodic time/phase and a non-periodic radii variation in the case of the real manipulation data. This knowledge can be incorporated in form of the specification of two associated latent dimensions: one periodic ($k_1(\cdot;\beta_1) = k_{\circlearrowleft}(\cdot)$) *temporal* and one non-periodic ($k_2(\cdot;\beta_2) = k_g(\cdot)$) *parameter* dimension. The objective function then consists of the reconstruction error and the penalty terms as denoted in Equation (5.6).

As proposed in the previous section, 3D Isomap embeddings are computed from the noisy training data $\mathbf{Y}$ (see Figure 5.6(e)). For this data, the choice of the neighbourhood parameter is very robust; in the depicted case, a value of $K = 10$ is used. Afterwards, atan2$^\star(\cdot)$ (5.7) has been used to retrieve a 2D latent initialisation for the UKR model (Figure 5.6(f)).

The evaluation focusses on the effect of different combinations of the inverse bandwidths $\beta_1, \beta_2$, and the penalty weightings $\lambda_{cseq}, \lambda_{pvar}$.

From the known structure of the generated toy data, initial guesses for good bandwidth parameters can be derived. In particular, all sequences consist of about 45 points which have an inter-point distance of $\frac{\pi}{45}$ when equidistantly distributed in an interval $[0;\pi]$. A good guess for $\beta_1$ can therefore be $\beta_1 = \frac{45}{\pi} \approx 14$. For similar reasons, $\beta_2$ can be estimated as $\beta_2 = \frac{1}{0.2} = 5$, since the inter-point distance of the training data in the second dimension is 0.2. The evaluation uses values around these landmarks, i.e. $\beta_1 \in \{7, 8, .., 14, .., 21\}$ and $\beta_2 \in \{3, 3.5, .., 5, .., 7\}$.

For $\lambda_{cseq}$ and $\lambda_{pvar}$, however, no such assumptions could be made, and thus, $\lambda_{cseq}$ and $\lambda_{pvar}$ are both evaluated for a broad range of possible values, i.e. $\lambda_{cseq}, \lambda_{pvar} \in \{0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}\}$.

For each tuple $(\beta_1, \beta_2, \lambda_{cseq}, \lambda_{pvar})$, 10 training runs with 10 noisy versions of the training data are conducted. Each run consists of 500 optimisation steps including leave-one-out cross-validation (one exemplary result can be seen in Figure 5.6(g)). Initial tests yielded the most promising results for $\lambda_{cseq} = \lambda_{pvar} = 1$ which provides a good starting point for the evaluation of $\beta_1$ and $\beta_2$.

Figures 5.7(a-d) depict the corresponding reconstruction errors for varying bandwidth parameters $\beta_1, \beta_2$.

Figure 5.7(a) shows the normalised mean square error (nMSE) between noise-free test data $\mathbf{Y}_T$ which describe the underlying true cylinder geometry and its UKR reconstructions $\mathbf{f}(\mathbf{g}(\mathbf{Y}_T))$. This error can be seen as a measure for UKR's ability to generalise to unseen data from the underlying structure.

Figure 5.7(b) shows the nMSE between $\mathbf{Y}_T$ and the reconstruction of its noisy versions $\mathbf{f}(\mathbf{g}(\mathbf{Y}_{Tn}))$. This error is used as a measure for UKR's robustness in representing the underlying structure and its ability to correct noisy input data.

The bias of $\mathbf{f}(\mathbf{g}(\cdot))$ towards the inner of the underlying structure, which is a known problem in original UKR, is depicted in Figure 5.7(c) for noisy training data $\mathbf{Y}_n$.

Figures 5.7(a-b) show a clear error dependency on $\beta_1$. The minimal errors yield the choice of $\beta_1 = 12$ in Figure 5.7(a) and $\beta_1 = 10$ in Figure 5.7(b), respectively. However, since the bias depicted in Figure 5.7(c) significantly increases with decreasing $\beta_1$, a value of $\beta_1 = 12$ yields a good compromise.

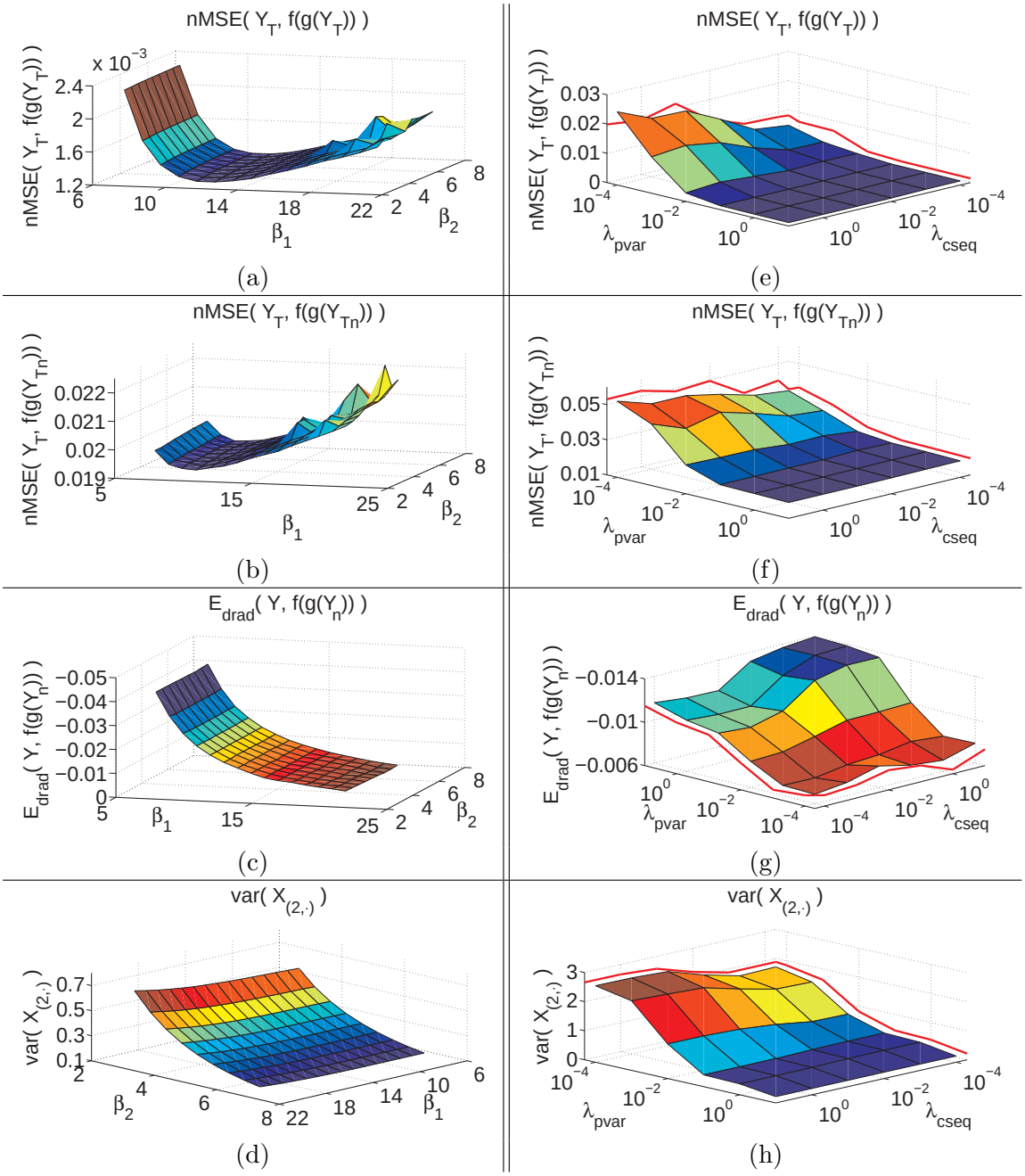



**Figure 5.7.:** Evaluation results **(a-d)** for $\lambda_{cseq} = \lambda_{pvar} = 1$ and **(e-h)**$(\beta_1, \beta_2) = (12, 4)$, red lines:$\lambda_{cseq}/\lambda_{pvar} = 0$.

As assumed before, there is no significant dependency on $\beta_2$ due to the free positioning of the latent parameters in the non-periodic dimensions (see Section 2.1.2). The effect of scaling in the non-periodic dimension in order to keep the error fixed is also visible in the evaluation results depicted in Figure 5.7: whereas the errors stay approximately constant for varying $\beta_2$ and fixed $\beta_1$ (Figure 5.7(a-c)), the variance in the non-periodic latent *parameter* dimension varies strongly for the same settings (Figure 5.7(d)). Small values of $\beta_2$ are therefore compensated for by a broader distribution of the latent variables yielding a larger

variance and vice versa.

In connection with the general considerations, i.e. that the inter-point distance of the training data in the second dimension is 0.2, a good choice for $\beta_2$ that yields robust kernel overlaps in average, but does not result in over-smoothing, constitutes a value of $\beta_2 = 5$ or slightly smaller. For the following, a value of $\beta_2 = 4$ has been used.

The inverse situation can be observed for $\beta_1$ in the periodic dimension. Here, the variance is approximately constant for all choices of $\beta_1$, and the errors vary largely. Consistent with the prior expectations, changing bandwidth parameters cannot be compensated for by moving the latent parameters, since the latent dimension is periodic and the available space is restricted to $[0; \pi]$.

The choice of $\beta_1$ is therefore more crucial. The error plots in Figure 5.7(a-c), however, reveal that similar considerations as described in the $\beta_2$-case also hold for $\beta_1$: the value of $\beta_1 = 12$ which has been identified as a good choice in the error analysis above corresponds to a value slightly below the average inter-point distance in the periodic dimension. It therefore constitutes again a bandwidth parameter that in average yields robust kernel overlaps without over-smoothing.

Using these considerations as a guideline for choosing appropriate bandwidth parameters, a good choice for the remaining evaluation is given by $(\beta_1, \beta_2) = (12, 4)$.

In order to analyse the effects of different penalty term weightings $\lambda_{cseq}$ and $\lambda_{pvar}$ under the condition of fixed bandwidth parameters $(\beta_1, \beta_2) = (12, 4)$, different combinations of $\lambda_{cseq}, \lambda_{pvar}$ are evaluated according to several error measures, as shown in Figure 5.7(e-h).

On the one hand, $\lambda_{cseq}$ weighs the penalty whose effect is to reinforce the correct chronological order of the latent variables. On the other hand, $\lambda_{pvar}$ is the weighting of the penalty that keeps the variance of latent sequence parameter values within single sequences low such that the representations of single sequences have approximately constant latent sequence parameters. The targeted combination of $\lambda_{cseq}$ and $\lambda_{pvar}$ therefore yields high values for both weightings (in order to realise the desired effects of the penalties), but results in small error values at the same time.

Figures 5.7(e-f) reveal that high values of $\lambda_{cseq}$ tend to negatively influence the reconstruction error. This negative effect can however be damped by high values of $\lambda_{pvar}$ and values $\lambda_{cseq} = \lambda_{pvar} = 1$ yield good results in both cases (e) and (f). The bias toward the inner of the true structure is rather large for this weighting choice as can be seen in Figure 5.7(g). However, since the variance in the sequence parameter dimension, which also constitutes an indicator for the individual sequence variance, dramatically increases for small weightings $\lambda_{pvar}$ as depicted in Figure 5.7(h), an increased bias needs to be accepted in order to realise the desired penalty effects.

For applications that exploit the clear, sequence-reflecting structure of the latent space, structure-related error measures are also important in addition to the pointwise nMSE discussed above. In particular, if the learning is successful, linear navigations through the latent space following a specific dimension are of special interest. Here, especially four different situations are relevant for applications:

1. Latent trajectories ${}^{\mathbf{r}}\mathbf{X}_{\mathbf{T}}$ following the latent time dimension (keeping the sequence parameters fixed) can be mapped into data space. Such mappings correspond to reproductions/synthesises of the represented sequence class for the underlying sequence parametrisation.

In the analysed case with a periodic time dimension, such latent trajectories correspond to a set ${}^{\mathbf{r}}\mathbf{X}_{\mathbf{T}}$ of latent test variables, that hold increasing latent time values and at the same time constant latent sequence parameters. Since the time dimension is periodic and the parameter dimension is non-periodic, such trajectories correspond to cross-sectional *rings* of the "latent space cylinder" and are marked with the superscript ${}^{r}$.
In the toy data case where the observation data represent a cylinder structure, the mappings of such latent trajectories are therefore targeted to result in cross-sectional rings of the cylinder without change in the height level in observation space.

2. Mappings of latent trajectories ${}^{\mathbf{l}}\mathbf{X}_{\mathbf{T}}$ following the latent sequence parameter dimension while keeping the latent time value fixed correspond to sequences in observation space that do not change the temporal aspect, but only the sequence parameters.
In the analysed case, such latent trajectories correspond to a set ${}^{\mathbf{l}}\mathbf{X}_{\mathbf{T}}$ of latent test variables, that hold increasing sequence parameter values with at the same time constant time values. Since these trajectories constitute straight *lines* in latent space, the test data is marked with the superscript ${}^{l}$.
In the toy data case, such mappings are therefore targeted to result in straight lines in observation space that are parallel to the cylinder axis.

3. Projections of sequences of observations ${}^{\mathbf{r}}\mathbf{Y}_{\mathbf{T}}$ that correspond to the class of represented sequences, i.e. that only change the temporal aspect of a sequence while keeping the set of sequence parameters fixed.
In the toy data example, due to the periodic representation of sequences, such observed sequences ${}^{\mathbf{r}}\mathbf{Y}_{\mathbf{T}}$ with fixed sequence parameters correspond to cross-sectional cylinder rings (again indicated by the superscript ${}^{r}$) with constant height level in observation space.
The projections of such observed sequences into latent space are targeted to result in trajectories corresponding to cross-sectional rings of the "latent space cylinder". This case therefore constitutes the inverse to Situation 1.

4. Projections of sequences of observations ${}^{\mathbf{l}}\mathbf{Y}_{\mathbf{T}}$ that correspond to a change of the sequence parameters without changing the temporal aspect.
In the toy data case, such observed sequences ${}^{\mathbf{l}}\mathbf{Y}_{\mathbf{T}}$ correspond to straight lines in observation space, parallel to the cylinder axis.
The projections of such sequences into latent space are targeted to result in straight lines with latent variables that hold increasing sequence parameter values with at the same time constant time values. This case therefore constitutes the inverse to Situation 2.

In order to analyse the method according to the behaviour of such mappings and projections, several additional errors are computed.

Figure 5.8(a) shows the evaluation results for latent trajectories ${}^{\mathbf{r}}\mathbf{X}_{\mathbf{T}}$ corresponding to cross-sectional rings of the "latent space cylinder" according to Situation 1. Since such rings are targeted to result again in cross-sectional rings in observation space, the normalised variance ("nVAR") orthogonal to the underlying ring in observation space of the mappings $\mathbf{f}({}^{\mathbf{r}}\mathbf{X}_{\mathbf{T}})$ provides a measure for the ring distortion and likewise for the distortion of the mapping in the analysed context.

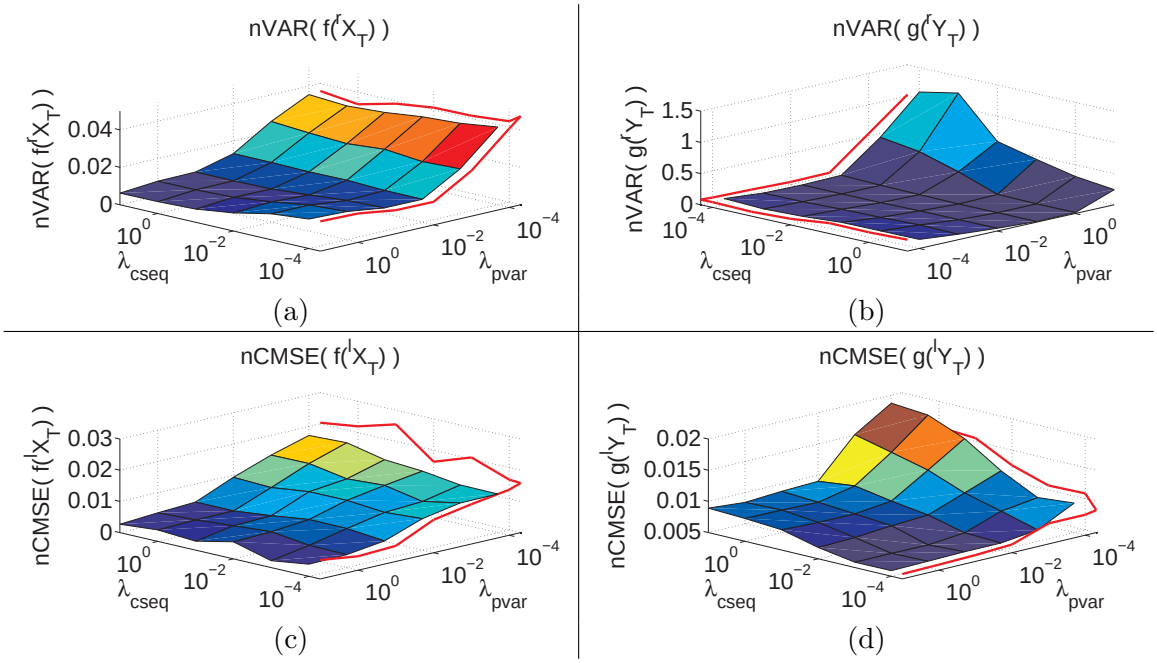



**Figure 5.8.:** Application errors for $(\beta_1, \beta_2) = (12, 4)$, Red line: $\lambda_{cseq}/\lambda_{pvar} = 0$

.

The plot uncovers that high weightings of $E_{pvar}$, that reduce general reconstruction errors as previously seen in Figure 5.7(e-h), only result in stable sequence mappings if $E_{cseq}$ is also weighted strongly. A good combination again constitute the values $\lambda_{cseq} = \lambda_{pvar} = 1$.

Figure 5.8(b) shows the evaluation results for the inverse situation, where observed sequences ${}^{\mathbf{r}}\mathbf{Y}_{\mathbf{T}}$ with fixed sequence parameter and constant height level in observation space are projected into latent space (Situation 3). For similar considerations, also the normalised variance orthogonal to the underlying ring in latent space of the projections $\mathbf{g}({}^{\mathbf{r}}\mathbf{Y}_{\mathbf{T}})$ are depicted. Again, the plot uncovers that only both high values for $\lambda_{cseq}$ and $\lambda_{pvar}$ result in stable projections. As before, $\lambda_{cseq} = \lambda_{pvar} = 1$ yield good results.

Figures 5.8(c-d) investigate the corresponding orthogonal situations, i.e. mapping line trajectories ${}^{\mathbf{l}}\mathbf{X}_{\mathbf{T}}$ with varying sequence parameter and fixed time value into observation space (Situation 2) and projecting sequences ${}^{\mathbf{l}}\mathbf{Y}_{\mathbf{T}}$ also with varying sequence parameter and fixed time value into latent space (Situation 4).

Since the latent time dimension is periodic in $[0; \pi]$, the normalised variance measure used before cannot be applied directly. Instead, a modified version is used, that also takes the periodic nature into account. Since the projections of ${}^{\mathbf{l}}\mathbf{Y}_{\mathbf{T}}$ are targeted to result in straight lines in latent space, the nMSE of the angular deviations from their means ("nCMSE") in the time dimension can be used as alternative measure for the line distortion, similar to the nVAR used before. If $\mathbf{y}_i = {}^{l}\mathbf{y}_{T,i} \in {}^{\mathbf{l}}\mathbf{Y}_{\mathbf{T}}$ and $\hat{\mathbf{x}}_i = {}^{l}\hat{\mathbf{x}}_{T,i} = \mathbf{g}(\mathbf{y}_i)$ with a time component

$x_{i,d_t}$ in the time dimension $d_t$, then

$$\delta_i = \mathrm{mod}_\pi(\| x_{i,d_t} - \langle x_{j,d_t} \rangle_j \|), \tag{5.8}$$

$$\Delta_i = \begin{cases} \delta_i & \text{if } \delta_i \leq \frac{\pi}{2} \\ \pi - \delta_i & \text{else.} \end{cases}, \text{ and} \tag{5.9}$$

$$e_{nCMSE} = \frac{1}{N\pi^2} \sum_i^N \Delta_i^2. \tag{5.10}$$

For the related case of projecting ${}^1\mathbf{X_T}$ into observation space, the corresponding version of $e_{nCMSE}$ for the periodicity in $[0; 2\pi]$ is applied.

Figure 5.8(c) depicts the nCMSE for Situation 2, i.e. the line trajectory mapped from latent into observation space. The evaluation reveals that high values for $\lambda_{pvar}$ in combination with high values for $\lambda_{pvar}$ yield good results. Also in this case, the choice of $\lambda_{cseq} = \lambda_{pvar} = 1$ can be used.

Figure 5.8(d) shows the nCMSE for Situation 4, i.e. projecting line trajectories in observation space which are parallel to the cylinder axis into latent space. High $\lambda_{cseq}$-weightings result in higher distortions of the projections. However, for the targeted high weightings of $E_{pvar}$, the negative effect of higher values for $\lambda_{cseq}$ still is in a reasonable region. A choice of $\lambda_{cseq} = \lambda_{pvar} = 1$ that has been proposed in the other situations does not yield the best behaviour in this context. The error increase is however only slight and is not important enough to overrule the positive effects of such choice in all other situations.

To sum up: The choice of the inverse bandwidth $\beta_1$ of the periodic dimension is a crucial issue. The average point distance of the training sequences in the periodic dimension can however be used as a guideline for a meaningful value: choosing $\beta_1$ to be slightly below the average point distance yields in average robust kernel overlaps without over-smoothing. Whereas the choice of the bandwidth parameter $\beta_2$ of the non-periodic dimension is not crucial, since suboptimal values can be compensated for by the learning, similar considerations as for the periodic dimension can be applied.

The penalty weightings $\lambda_{cseq}$ and $\lambda_{pvar}$ need to be set to high values in order to have the penalties influence the learning result. Indeed, in the case of the toy data evaluation, both high weightings resulted in good results for both the general reconstruction errors as well as for the application-related errors.

### 5.3.4. The real data case

The main target of the extensions described in the Section 5.3.1 is to generate manifolds similar to the *Manipulation Manifolds* resulting from the manifold construction scheme described in Section 5.2, but in a more automatic manner: specific movement parameters – and especially the advance in time – are explicitly represented by specific and distinct manifold dimensions. The manifold structure itself therefore represents knowledge about the manipulation. A manipulation controller which exploits this simplified structure can easily perform manipulation movements by just straightly navigating through the manifold's latent space.

Incorporating the new UKR extensions for chronologically ordered data sequences and the results of the analysis described in the previous section into the manifold training

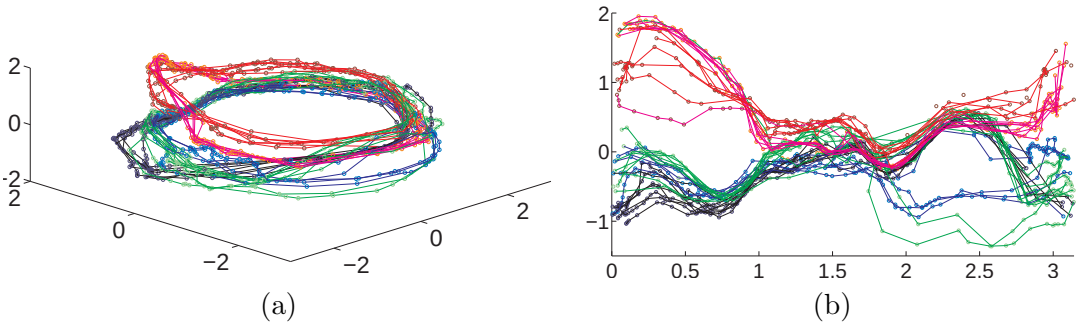



**Figure 5.9.:** Structure of the cap turning hand posture data. Different colours encode different cap radii ($r = 1.5cm$ (black), $2cm$ (blue), $2.5cm$ (green), $3cm$ (magenta), and $3.5cm$ (red)). Connected points illustrate neighbouring hand postures within the sequences. **(a)** 3D Isomap embedding (K=10) of the hand posture data. The periodicity of the cap turning movement is clearly reflected in the cylinder-like embedding structure. **(b)** 2D unfolding of the cyclic structure of the Isomap embedding in (a) using atan2$^\star$. This form is used as initialisation of the UKR latent parameters.

mechanism enables a *semi-supervised* learning scheme for such *Manipulation Manifolds*. The manual constructions of the UKR latent parameters can be substituted by the new scheme.

Since UKR requires a fixed number of latent dimensions, the adequate amount of dimensions needs to be specified beforehand based on prior knowledge about the structure of the underlying training data. In the case of the manipulation data – corresponding to the constructed version of the *Manipulation Manifolds* – a two-dimensional latent representations ($q = 2$) of the 24-dimensional hand posture data is a good choice. One dimension corresponds to the cap radius and the other to the temporal advance within the cap turning movement. Taking the new features of UKR for periodic data sequences into account, the dimension-specific kernels can be chosen as $k_1(\cdot) = k_{\circlearrowleft}(x; \beta_1)$ (5.3) as periodic kernel for the periodic temporal dimension and $k_2(\cdot) = k_g(x; \beta_2)$ (5.2) as non-periodic kernel for the non-periodic radius dimension.

For initialising the model, a 3D embedding of the data using Isomap is computed. The neighbourhood radius is set to $K = 10$ in this case, but the results for the data are very robust against different choices of $K$. The resulting embedding is visualised in Figure 5.9(a). Since standard Isomap cannot directly embed into a periodic space, the extra dimension is necessary to preserve the local structure of the data. However, it is straightforward to detect the 2D subspace containing the periodicity, and to map these coordinates to an angular representation. In this case, atan2$^\star$ (5.7) has been applied, leaving the non-periodic dimension unchanged (see Figure 5.9(b)).

Following the parameter selection hints determined from the observations in the toy data evaluation in the previous section, the parameters $(\beta_1, \beta_2, \lambda_{cseq}, \lambda_{pvar})$ of the UKR model are chosen as $(10, 4, 1, 1)$. Five different cap turning sequences are considered as training data, each consisting of all available training data for one specific cap radius.

On the basis of this training data, the gradient-based UKR optimisation of the extended objective function (5.6) is performed for 600 iterations. Figure 5.10 visualises the different stages of the latent parameters during the training. Figure 5.10(a) depicts the utilised initialisation of the latent parameters. Figure 5.10(b-e) show their development after 5, 10, 15, and 25 optimisation steps. Already in this early stage of the training, the representations of the single sequences are flattened in their radius dimension. This early effect results

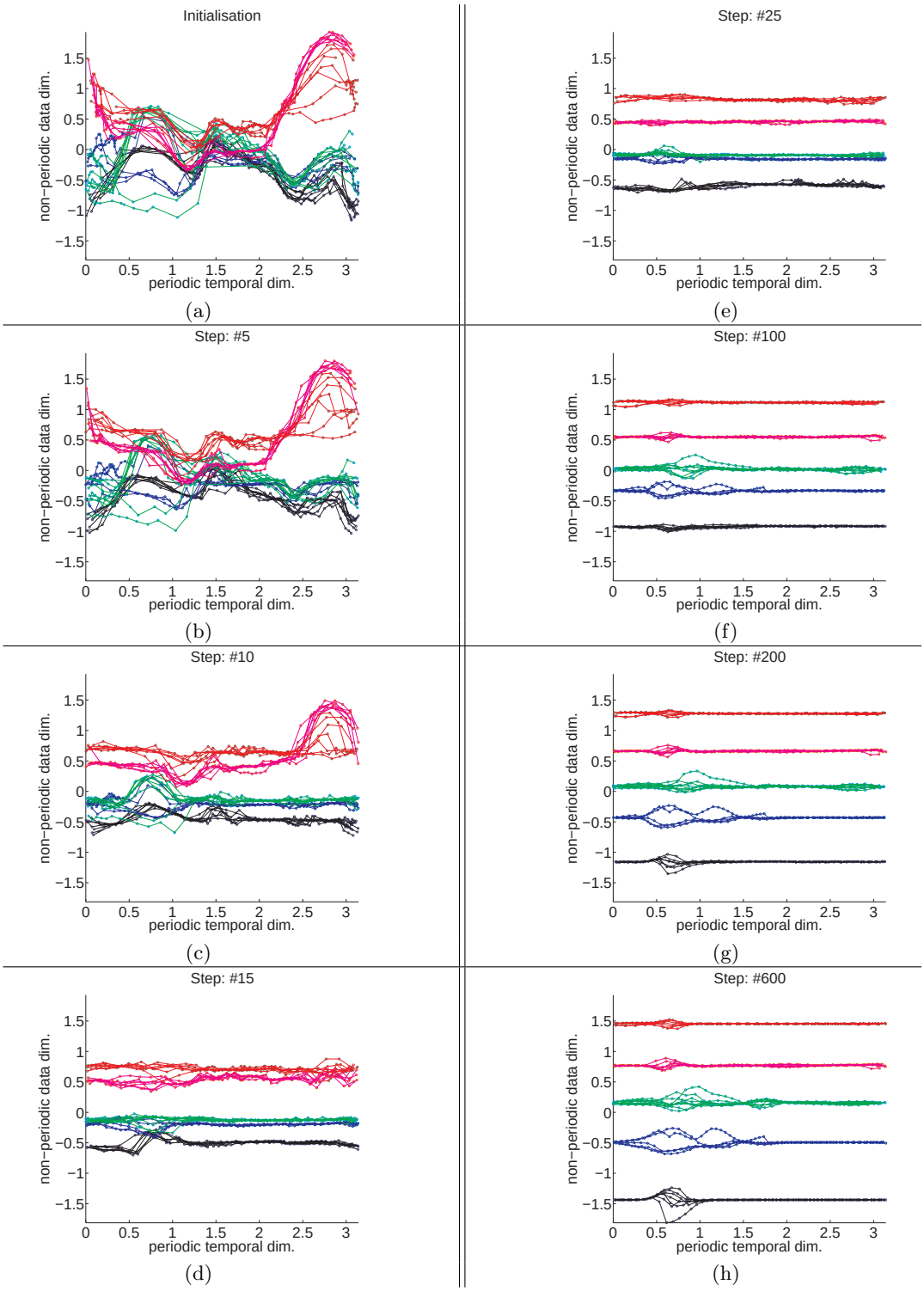



**Figure 5.10.:** Development of UKR latent variables during training (Colours as in Figure 5.9). **(a)** Initialisation with atan2$^\star$-mapped Isomap-embedding. The horizontal direction corresponds to the periodic temporal latent UKR dimension, the vertical to the non-periodic latent data dimension. **(b-e)** after 5, 10, 15 and 25 optimisation steps: intra-sequence variances in latent data dimension are reduced, the separation of different sequences is not yet finished. **(f-g)** after 100 and 200 steps: sequence representations are now correctly ordered, flat in the (vertical) data dimension and clearly separated. Larger inter-sequence distances yield more intra-sequence distinctions (inducing larger intra-sequence data variances). **(h)** after 600 steps: training result.

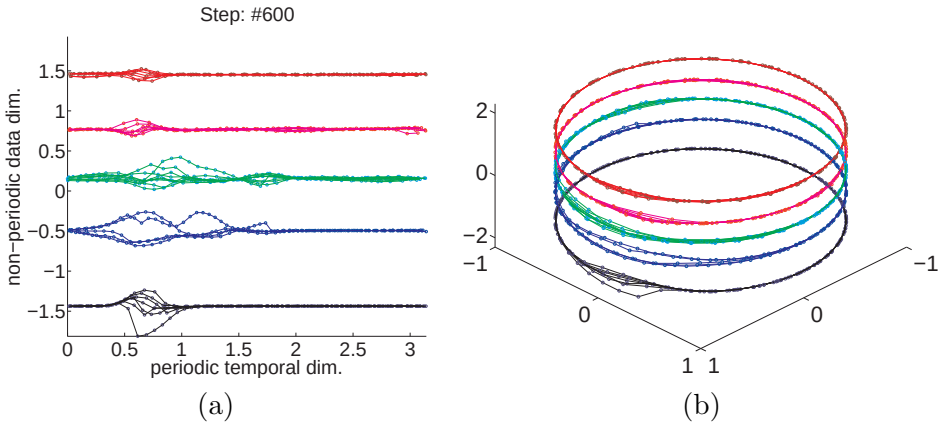



**Figure 5.11.:** Training result. (Colours as in Figure 5.9). **(a)** 2D plot of the latent parameters. The horizontal direction corresponds to the cap radius (*data*) dimension, the vertical to the periodic *temporal* latent dimension (period: $[0; \pi]$). **(b)** 3D mapping of (a). Here, sin/cos are used to generate the 2D-circular representation of the 1D-angular temporal latent dimension.

from the still tightly packed sequences pushing each other apart and the regularisation of the variance in the radius dimension of the single sequences using the parameter variance penalty (5.5).

The order of the sequence representations in radius direction corresponds to the correct order of the radii (from bottom to top): black ($r = 1.5cm$), blue ($2cm$), green ($2.5cm$), magenta ($3cm$), and red ($3.5cm$). However, the separation of the sequences is not yet finished – especially the sequences for $r = 2cm$ and $2.5cm$ (blue and green) are very close to each other and partly overlap. Indeed, the distinction is intensified in the following training (see Figure 5.10(f-g): after 100 and 200 steps, respectively) and finally very explicit after 600 steps, as depicted in Figure 5.10(h). However, the larger *inter*-sequence distances yield more space for *intra*-sequence distinction as well. The variance in the radius dimension of the single sequence representations therefore grows with progressing sequence separation and results in partly non-flat sequence representations.

### 5.3.5. Discussion

The resulting latent parameters of the UKR training for periodic data sequences are visualised in Figure 5.11. The latent structure reflects the initially targeted characteristics: the representation of the single sequences of training data are well separated from each other and have low variance in their data (radius) dimension.

However, from the application point of view, also the task related abilities of the resulting manifold are of special interest. The main focus of this evaluation therefore lies on the manifold's capability to represent and reproduce the underlying movements (or manipulations) and to synthesise new unseen motions. The very fine aspects however depend on very subtle details of the object, the object contacts, and the object-hand relation, which cannot be recorded by means of the available devices and therefore not represented either. The manifold basis therefore rather constitutes a guideline of how the fingers need to be coordinated. An evaluation on the basis of reconstruction errors is therefore of minor interest in this context.

Figure 5.12 visualises the training result in form of hand posture pictures. The postures correspond to the data space images $\mathbf{f}(\mathbf{x}; \mathbf{X})$ of regularly sampled positions $\mathbf{x}$ in latent

space. The bottom row corresponds to the minimal value of the latent radius dimension (which is also the smallest cap radius in this case) and the top row to the maximal value. In horizontal direction, the temporal dimension – periodic in $[0; \pi]$ – is sampled in steps of $\frac{1}{10}\pi$, whereas the last column is $\frac{9}{10}\pi$. Hence, the next step would be $\frac{10}{10}\pi$ which is equivalent to the first column ($= 0$) due to the periodicity of the horizontal dimension.

The represented movement consists of different phases:

1. columns 0 to $\frac{2}{10}\pi$: the hand is opened and moves in the direction of the little finger (backward movement),

2. columns $\frac{2}{10}\pi$ to $\frac{4}{10}\pi$: the fingers close depending on the associated cap radius (closing movement),

3. columns $\frac{4}{10}\pi$ to $\frac{7}{10}\pi$: the closed hand moves in the direction of the thumb and rotates around the cap position (turning movement).

4. Afterwards, the hand opens again (columns $\frac{7}{10}\pi$ to $\frac{9}{10}\pi$) and the motion smoothly transitions back to the beginning in column $0(= \pi)$.

Whereas the hand postures in the open hand phase are very similar for the different radii, they significantly differ in the turning phase (columns $\frac{4}{10}\pi$ to $\frac{7}{10}\pi$). In this phase, the fingers had contact with the cap in the training data generation and the different levels of hand closure around the cap position are clearly visible.

A comparison of the new learned Manipulation Manifold depicted in Figure 5.12 with its constructed predecessor shown in Figure 5.4 reveals that the latent structures of both are very similar. Since the radius values were not included in the training data for the automatic learning, the absolute correspondence between latent radius value and real radius is not provided anymore. However, since the exclusion of this data constituted one of the goals of the automatic learning scheme and the relative radius information can still be used for an adequate motion control (which will be shown in the following), the learning result constitutes a success. It is important to notice that the new latent structure is periodic in the temporal dimension and does therefore not suffer from the manifold border effects witnessed with the constructed version.

As result of the learning (or construction), the horizontal dimension corresponds to the temporal aspect of the movement and the vertical dimension describes the cap size as motion parameter. It forms a representation of the movement of turning a bottle cap for different cap sizes that fulfils the goal of fitting to the desired simple control strategy: the represented movement can be produced by mapping a linear trajectory that follows the time dimension in latent space into hand posture space.

This characteristics is realised by distorting the underlying geometry of the data space. Such strategy is necessary since the different parts of the represented motion feature different degrees of variations in the underlying hand postures: Those parts of the movement which are independent of the cap radius hold hand postures which are very similar for all radii. One example is the backward movement of the hand which is depicted in Figure 5.12(columns 1-3). The corresponding latent representations therefore have to be pushed away from each other in order to span the same latent radius range as the parts of the motion which are highly radius-dependent. These parts, in contrast, correspond to hand postures that are strongly dissimilar.

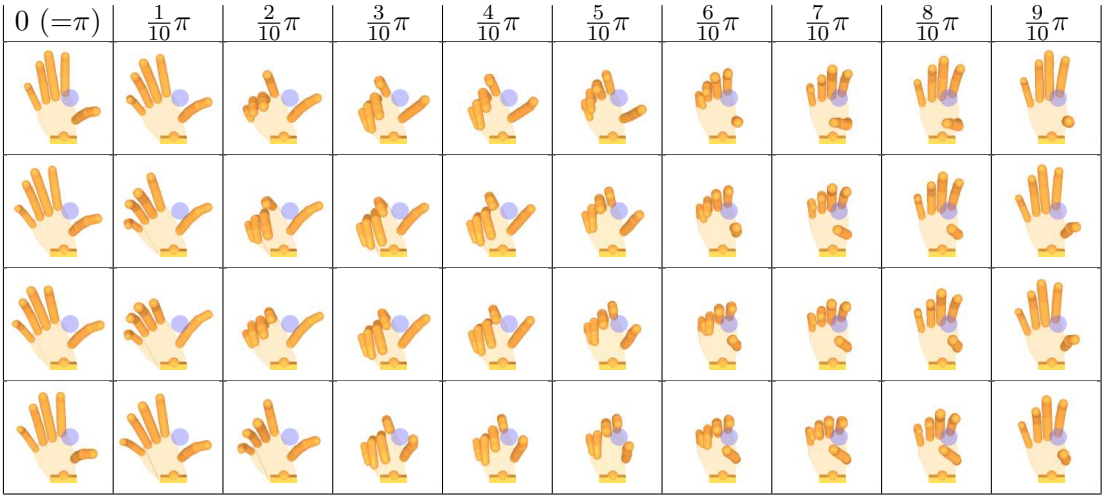



**Figure 5.12.:** Visualisation of the training result in the hand posture space. The depicted postures correspond to the mappings $\mathbf{f}(\mathbf{x};\mathbf{X})$ of regularly sampled positions $\mathbf{x}$ in the trained latent space. In each picture, a bottle cap of the radius $r = 1.5cm$ is shown as comparison aid. The latent radius dimension is mapped onto the vertical direction, the latent temporal dimension onto the horizontal direction of the depicted grid. The temporal dimension is periodic in $[0;\pi]$.

In consequence, the similar parts would collapse to thin regions in latent space in a purely unsupervised learning approach like the original UKR. This however would render the targeted control scheme impossible.

Indeed, whereas the distortion is beneficial for the production of motions, it poses some problems for the inverse direction, i.e. projecting hand posture sequences into latent space. In that case, whereas the temporal information is robust, the projection is strongly non-robust in the parameter (radius) dimension for the described radius-independent parts. Here, the corresponding hand postures are fairly similar for the whole range of latent radii for a specific point in time and thus, the result of $\mathbf{g}(\mathbf{y})$ can heavily vary for small changes of $\mathbf{y}$.

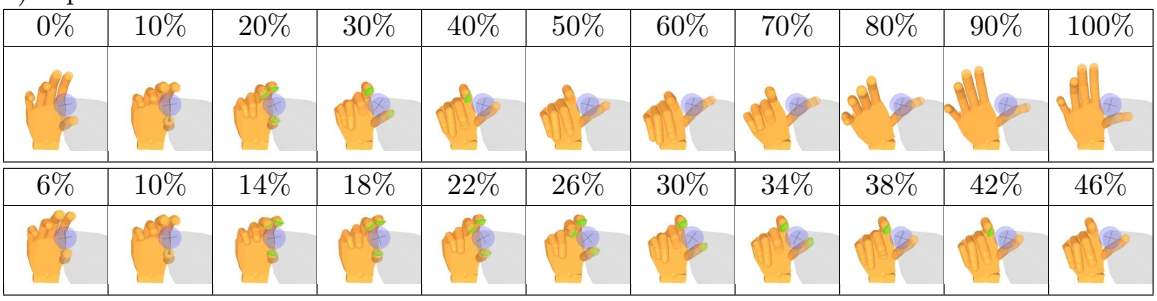
## 5.4. Open-Loop Control by Latent Space Navigation

The basic idea of the manifold generation in the way it is presented in the preceding sections was to create a robust representation of manipulation movements that facilitates to perform the associated manipulation in later applications. Due to this design, the reproduction of the represented movements can be easily done by exploiting the manifold structure.

For the reproduction, one desired latent radius value has to be initially chosen and then kept fixed. In the case of the manifold construction, since designed accordingly, this value corresponds to the actual radius. In the learning case where the radius dimension is subject to UKR optimisation as well, however, the absolute radius value cannot be chosen directly and has to be evaluated e.g. by visual inspection. For the manipulation of a cap, this is no disadvantage, as will be described in the following.

Given an adequate initial latent position, the represented motion can be generated by iteratively increasing the time value of the latent parameters. By mapping the resulting latent radius/time combination from latent space into observation space, a corresponding hand posture series can be obtained and actuated by the robot hand or its simulated model.

a) cap radius $r = 2cm$




b) cap radius $r = 2.75cm$

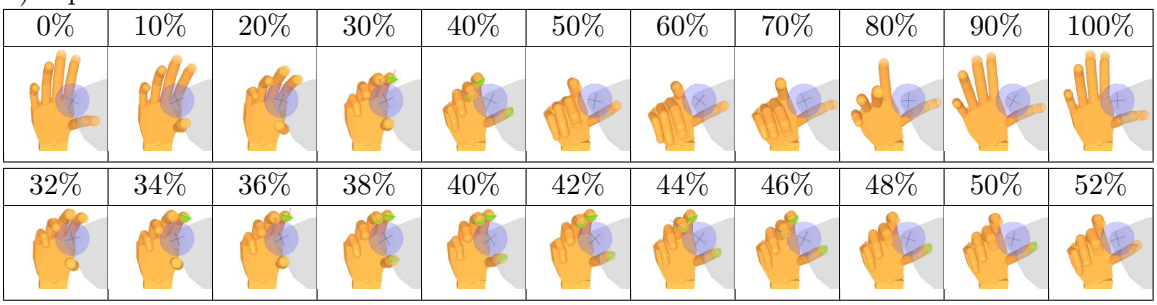



**Figure 5.13.:** Application results: hand posture sequences generated by applying the presented *Manipulation Manifolds*. The sequences depict intermediate hand postures during the turning movement for radii a) $r = 2cm$ and b) $r = 2.75cm$. In each case a) and b), the first rows visualise the complete movement cycles ($0\% - 100\%$) and the second rows focus on the periods with object contact. Green cones visualise contact friction cones. The rows have different time scales.

In the constructed version, which does not feature periodic latent dimensions, the latent time component has to be reset to 0 when reaching the manifold border.

## 5.5. Feedback-Initialised Open-Loop Control: The Bottle Cap Example

For the generation of manipulations, one simple algorithm works in a very similar manner as the open-loop navigation described in the last section. Instead of manually selecting a cap radius, the bottle cap can be grasped first. The resulting hand posture can be projected onto the manifold and the radius component of the resulting latent parameter can be used as radius initialisation. After having identified the appropriate radius value, the motion control can be again performed in open-loop manner, as described above. Although, in this simple approach, neither contact occurrences nor contact forces are taken into account, a rudimentary manipulation can be successfully realised due to the simple design of the manifold.

### 5.5.1. Manipulation initialisation with Experience-based Grasping

For the grasping of the bottle cap, the experience-based grasping algorithm described in Section 4.1 can be applied. In this case, the *Manipulation Manifold* can be directly used as experience basis. Since the grasping approach originally has been designed to work with *Grasp Manifolds* consisting of hand postures that represent final grasp postures, the *Manipulation Manifold* provides a large amount of hand postures which do not fit to the original

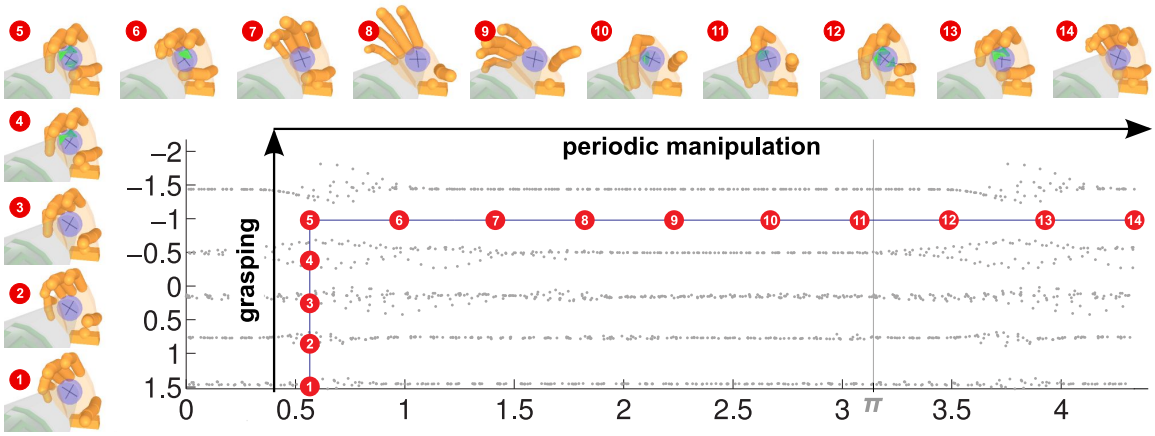



**Figure 5.14.:** Visualisation of an exemplary grasping/manipulation sequence using the simplified manipulation controller on the trained manifold. The graph depicts the distribution of the UKR latent parameters (the horizontal latent time dimension is periodic in $[0; \pi]$ and the axis direction of the radius dimension is reversed). The hand pictures show 14 intermediate postures during the manipulation movement corresponding to the marks '1' to '14' in the graph. The arrows ("grasping", "periodic manipulation") represent the two very simple stages of the controller: a) grasp the cap by navigating through the manifold's latent space from a fix starting point (pic. 1) in the direction of decreasing radius (pic. 2-5) until thumb, fore finger and middle finger have contact (pic. 5) and then b) perform the turning movement by navigating in the orthogonal temporal latent dimension (pic. 5-14). Since the temporal dimension is periodic, the turning movement can be reapplied (with smooth transition from one run to the next) by just further increasing the temporal value.

requirements. The algorithm nevertheless achieves in many cases to find the appropriate final grasp. Due to the many non-grasp postures in the manifold, the approach of the fingers however is not optimal.

Figure 5.13 depicts two sequences of intermediate hand postures during a bottle cap manipulation for two different radii (a) $r = 2cm$ and (b) $r = 2.75cm$ applying the described approach. The first rows each depict one whole movement cycle ($0\% - 100\%$) whereas the second rows each show in detail the period of object manipulation in which the fingers have contacts to the bottle cap. Both manipulations are successful in terms of rotating the cap, but as depicted in the second rows of Figure 5.13(a) and (b), the period of the very manipulation where the cap is actively rotated is much longer for $r = 2cm$. In the first case, the active manipulation takes place between 14% and 34% with at least two opposing contact fingers, and up to 42% with at least one finger contact remaining. In the second case, this period only lasts from 36% to 46% with opposing contact fingers, and up to 50% with one finger contact.

### 5.5.2. Manipulation with simplified initialisation

A more robust and at the same time less complex manipulation controller performs the radius adaptation in a simplified manner (see Figure 5.14). Instead of using the experience-based grasping, the algorithm starts in an initial hand posture which is associated with a fixed latent position on the manifold. This latent position lies on the 'maximum radius' border of the latent space in a temporal position where the fingers have contact with the cap (see Figure 5.14, Position 1). This initial position needs to be specified beforehand by visual inspection or – if present in the training data – by analysing the finger contacts

for different latent time values. If thoroughly examined, this position can serve as general starting posture for manipulations with all covered cap radii. The motion controller then is subdivided into two different phases of orthogonal, straight navigations through the latent space:

1. The first phase corresponds to grasping the cap. It is realised by a straight navigation in direction of decreasing radii following the radius dimension. The phase is visualised in Figure 5.14 as a vertical arrow and the trajectory part from Position 1 to Position 5. The corresponding closing movement of the fingers is also depicted in the figure as single frames. It continues until thumb, fore finger and middle finger have contact (this is the case in Position 5). The resulting latent position yields an appropriate latent radius value for the subsequent manipulation movement.

2. As soon as this simple grasping method succeeds, the controller transitions to the manipulation phase, in which the previously adapted radius is constant. The manipulation movement is performed by navigating through the latent space following the temporal dimension (Figure 5.14, Positions 5 to 14). Since in this example, the latent space is periodic in this dimension, the cap turning movement controlled this way can be repeated several times by just further increasing the temporal latent value.

### 5.5.3. Discussion

The feedback-initialised open-loop controllers presented in this section both follow the same, basic approach: in a first phase, the bottle cap is grasped in order to get a valid hand posture that holds information about the cap radius. By projecting this hand posture onto the manifold and therefore into the latent space, the resulting latent radius value is used as parametrisation of the following manipulation movement. The very manipulation then just consists of following the temporal dimension of the latent space in an open-loop fashion.

The first approach constitutes a possible combination of the manipulation with the experience-based grasping approach presented earlier. Indeed, the manipulation is a consequent extension of the grasping approach: whereas the grasping algorithm pulls the posture of the actuated hand onto the manifold (the hand is actuated such that the resulting posture corresponds to the one point in the manifold that corresponds to the grasping hand posture), the manipulation realises the subsequent navigation through the manifold.

The second approach exploits the simple structure of the manifold already for the grasping phase. Initially, the hand posture is also pulled onto the manifold, but instead of directly actuating the grasp posture, the initial *pregrasp* posture at the manifold border is actuated. From this pregrasp, the very grasp motion is performed by a navigation through the latent space, similar to the subsequent manipulation, but in orthogonal latent direction.

The open-loop manipulation produces the cap turning motion that is represented in the manifold for the identified latent radius value. Since no further feedback like contact positions or forces are considered, it may however not perfectly fit to the actual manipulation context. Indeed, it rather corresponds to humans that try to perform a cap turning motion with no cap present. They also perform a similar motion, but only in combination with the sensation of the real cap, the manipulation motion gets realistic and effective.

The incorporation of feedback not only for the initialisation, but also during the control is therefore an important issue. In the next section, a method of using Structured UKR manifolds is presented that enables such incorporation of sensory feedback.

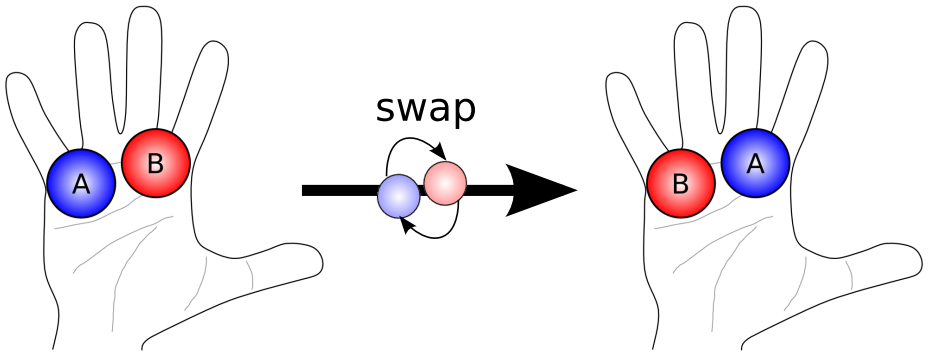



**Figure 5.15.:** Schematic visualisation of the Chinese health balls swapping. For the implementation on the real robot hand, the movements has been restricted to the four fingers excluding the thumb.

## 5.6. Closed-Loop Control: The Chinese-Health-Ball Example[5]

In the introduction to this chapter, the work of Oztop et al. (2006) has been used to motivate the need for a more general and sophisticated representation of manipulation movements. Oztop et al. (2006) indeed achieved to directly record robot-appropriate training data for the task of swapping Chinese health balls as described in Figure 5.15 and represent the motion in an open-loop fashion such that the robot was able to perform the same ball swapping task without human guidance. However, since the proposed controller framework does not allow for the incorporation of any sensory feedback, it is limited to actuating the represented motion in a feed-forward manner and is therefore not able to react to unforeseen situations.

This section describes a method to derive a closed-loop controller for the same ball swapping task taking the current positions of the balls into account and using them as control parameters for the underlying hand motion. In this way, the method overcomes the former shortcomings and realises an adaptive manipulation control.

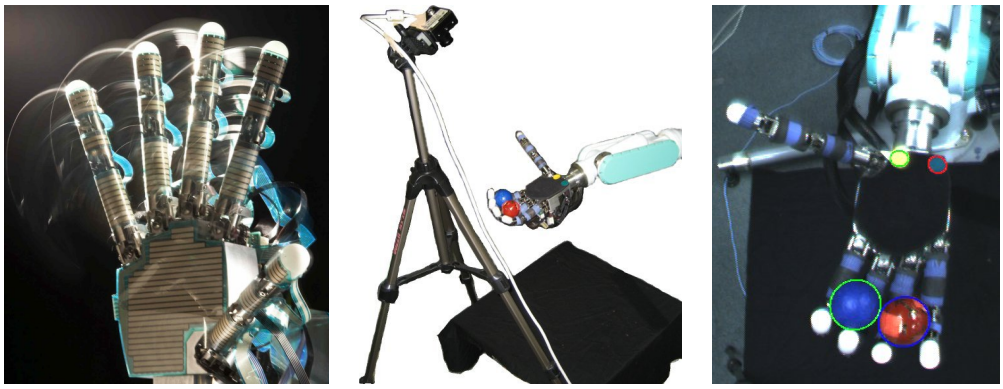
In order to represent the motion data together with their corresponding manipulation states – in the presented case the 2D positions of the balls in the palm plane – an intuitive way of using Structured UKR manifolds as a basis for a closed-loop control framework is described. In combination with the inherent characteristics of the Structured UKR manifold, the representation then lends itself to a simple and robust feedback control scheme that allows for sensory-driven motion control.

### 5.6.1. The Robotic Setup

The robotic hand used for the implementation of the proposed controller framework is the five-fingered 16 degrees of freedom Gifu Hand III (see Section 3.1.2 for a detailed description) consisting of a thumb and four fingers (Figure 5.16(a)). While the thumb provides four independent joints resulting in four DOF, the fingers only have three DOF since - in each case - the two distal joints are coupled.

For the ball swapping task, the Gifu Hand was mounted on a PA-10 robot arm (Mitsubishi Heavy Industries) in order to adjust the orientation of the Gifu Hand in a similar way as

[5]The work presented in this section is based on the work developed by the author in collaboration with Erhan Oztop during a research stay in 2009 at the Computational Neuroscience Laboratories (CNS) at the Advanced Telecommunications Research Institute International (ATR), Kyoto, Japan, in Prof. Dr. Kenji Doya's group.



(a) (b) (c)

**Figure 5.16.: (a)** The Gifu Hand III. **(b)** The ball tracking setup. The camera for recording the pictures used by the colour blob tracker is positioned above the hand viewing in the direction of the palm plane. **(c)** Exemplary picture of the colour blob tracker camera shown in (b). Tracked blobs (ellipses) are marked by surrounding lines: the two balls with green and blue ellipses; the reference points with red and green.

in (Oztop et al., 2006) (see Figure 5.16(b)). In addition, a camera was placed above the scene and directed towards the palm of the hand (see Figure 5.16(b)). Using the camera pictures, a colour blob tracker[6] provides 2D positions in the palm plane of the two balls relative to the reference blobs near the wrist (see Figure 5.16(c)). This visual evaluation of the ball states substitutes a more intuitive solution of using a tactile sensing system which was not available in the Gifu Hand.

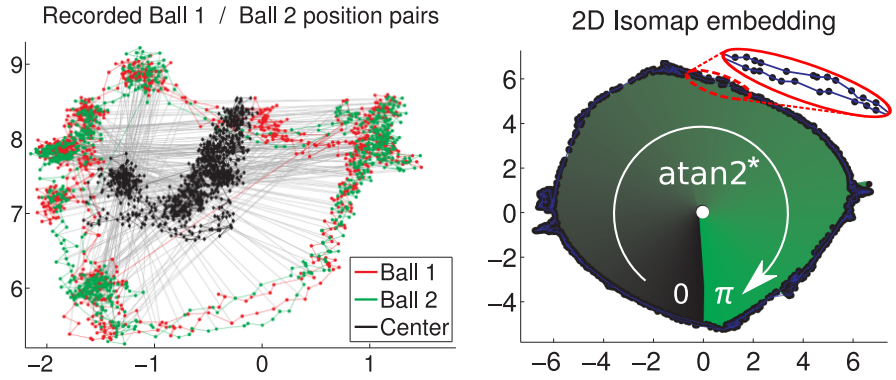
### 5.6.2. Training of the Structured UKR model

As a basis for the training of the Structured UKR manifold, a sequence of hand postures has been recorded together with the corresponding ball positions during four successful cycles of the ball swapping manipulation generated by the original open-loop controller from Oztop et al. (2006). Since the recorded data was produced by fixed open-loop control, it does not provide variations in the finger movements between the cycles. The corresponding ball positions however can differ from one cycle to the next which thus enriches the data basis. Figure 5.17(a) shows the recorded ball pair trajectories (each position normalised such that the reference points shown in Figure 5.16(c) are $(-1,0)^T$ and $(1,0)^T$, respectively) and gives an impression of the symmetry in their movements.

For the design of the manifold, whose latent control space dimension has to be specified beforehand, this data cannot be expected to provide enough variations from the demonstrations allowing for a specific latent dimension covering different *versions* of the underlying manipulation movement. Thus, the latent space has been restricted to only represent the (one-dimensional) temporal aspect of the manipulation ($q = 1$).

For the task of turning a bottle cap detailed in the previous section, the latent control parameters $\mathbf{X}$ contained also the radius of the bottle cap. In that case, the observations $\mathbf{Y}$ only consisted of the observed finger joint angles. In this new representation which is aimed at feedback control, the ball positions cannot be treated as dedicated control parameters

[6]The author would like to thank Aleš Ude for providing the colour blob tracking software and for his kind introduction it.




(a) (b)

**Figure 5.17.: (a)** Recorded trajectories of ball 1 (red) and ball 2 (green). Corresponding ball 1 / ball 2 positions are connected by grey lines. The black points mark the trajectory of grey line centre points. Ball positions are normalised such that the wrist reference points (see Figure 5.16(c)) are at $(-1, 0)^T$ and $(1, 0)^T$. **(b)** 2D Isomap embedding of all composed hand posture/ball positions observations. The periodic nature is clearly visible. Closer inspection reveals that the underlying four ball swapping motions are embedded as two cycles resulting from the longer period in the ball position dimensions (see text for details). Applying the atan2$^\star$ on the data results in angular positions in $[0, \pi]$ (shown by the green colour gradient) which can be used as initialisation for the UKR latent parameters.

in the latent space since the positioning of the latent parameters is subject to a gradient-based optimisation during the training. Thus, they are rather relative values and may largely change with the development of the optimisation process.

Such a strategy can therefore not be applied in the case of the targeted task of representing the ball swapping in a way that inherently facilitates a closed-loop control. Since these positions will serve as control parameters directly specified by sensory feedback, a more fixed relationship between observed hand postures and the corresponding ball positions is required.

In particular, the association of specific hand postures to the corresponding ball position pairs needs to be maintained in the learning phase. Consequently, these quantities may not be separated from each other on the data representation level. This however was the case in the cap turning example where the hand postures have been treated as observations and the ball positions as latent parameters.

The ball positions are therefore kept as part of the observations by creating an 'observation vector' $\mathbf{o}$ consisting of the 16-dimensional hand posture $\mathbf{\Theta}$ and the two-dimensional ball positions $\mathbf{p}_1$ and $\mathbf{p}_2$ : $\mathbf{o} = (\mathbf{\Theta}^T, \mathbf{p}_1^T, \mathbf{p}_2^T)^T$.

Including the ball positions in the observations however comes with an unfavourable side effect: the period of the ball movement – which consists of two ball-swappings in order to bring the balls back to their initial positions – spans over two periods of the hand movement. Figure 5.17(b) depicts a two-dimensional Isomap embedding of the recorded training data of the four ball swappings (or bringing the balls back to their initial positions twice).

A closer inspection of the embedding reveals that the data is embedded in only two cycles. The combination of ball and hand movements to observation vectors consequentially yields data which are periodic over the whole period of the *ball* movement. This corresponds to two periods of the *hand* movement.

The net effect of these characteristics of the composed observations is that the basic

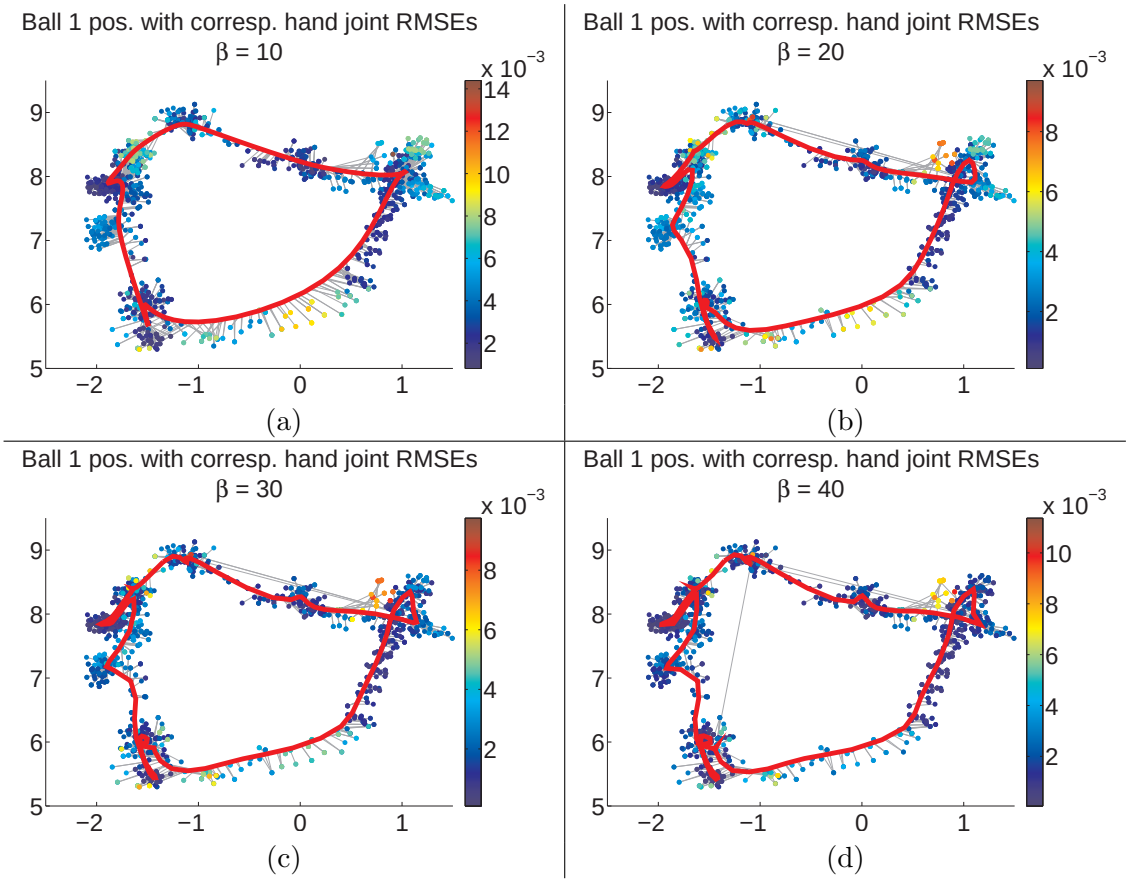



**Figure 5.18.:** Visualisations of the data and four trained models from the perspective of the ball positions. Depicted are the normalised positions of Ball 1 (coloured dots; normalised such that the wrist reference points – see Figure 5.16(c) – are at $(-1,0)^T$ and $(1,0)^T$) in the recorded data and the Ball 1 trajectory as represented in the trained models (red line) for different bandwidth parameters $\beta$: **(a)** 10, **(b)** 20, **(c)** 30, **(d)** 40. The colours of the dots encode the RMSE between recorded and corresponding represented 16 dimensional hand posture as visualised in the colour legend. The gray lines depict the projection errors for the Ball 1 positions resulting from reproducing the recorded composed observations: $\hat{\mathbf{o}} = \mathbf{f}(\mathbf{g}(\mathbf{o}))$.

hand movement is represented twice in the manifold. This results in two (slightly different) movements for swapping ball 1 or ball 2, respectively.

In order to train the manifolds, the method for learning Structured UKR manifolds as described in Section 5.3 has been applied. The set of observations consisted of 1757 composed vectors comprising 16-dimensional[7] Gifu Hand postures (finger joint angle vectors) and two two-dimensional ball positions resulting in total in $d = 20$ dimensions. For the latent space, the dimensionality has been set to $q = 1$ which corresponds to the latent *time* or phase dimension. To take the periodicity of the manipulation into account, the periodic kernel $k_1 = k_{\circlearrowleft}$ (5.3) has been used.

The latent initialisation is realised using the atan2$^\star$-mapping (5.7) of the 2D Isomap embedding of the training data (see Figure 5.17(b): white circular arrow and black to green colour gradient). As sequence-order penalty weighting, the default value $\lambda_{cseq} = 1$ is

[7]Since the thumb is not active in our manipulation, only 12 of the 16 dimensions have effectively varying values.

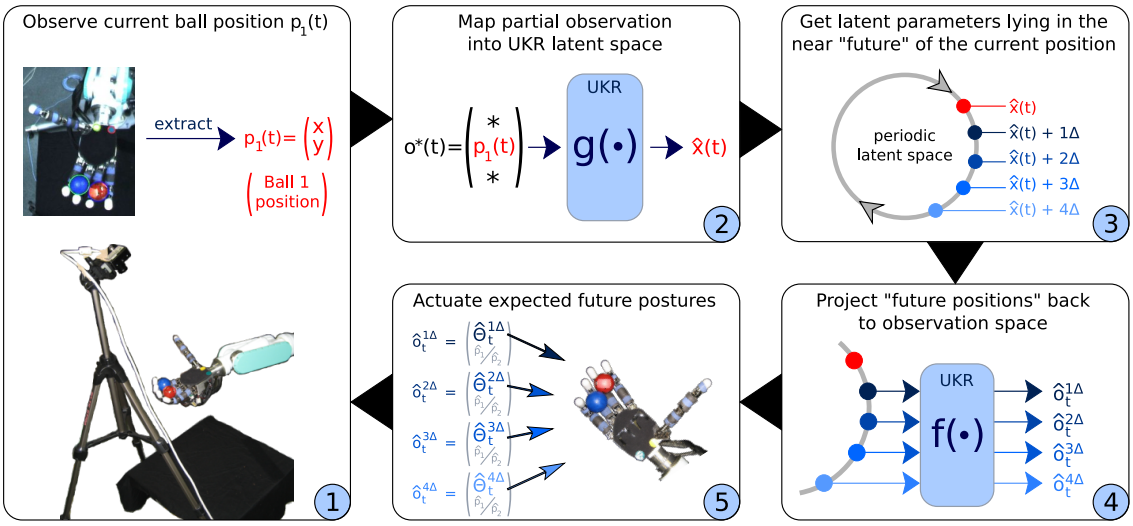



**Figure 5.19.:** Schematic overview of the different steps of the feedback control using Structured UKR manifolds (the corresponding step number is marked in the bottom right corner of each box). **Step (1)**: The current manipulation state in the form of the ball-1-position $p_1(t)$ is acquired using the camera with the colour blob tracker in order to extract the two-dimensional position (relative to the reference points near the wrist). **Step (2)**: The ball-1-position $p_1(t)$ – treated as a partial observation of the form $\mathbf{o} = (\mathbf{\Theta}^T, \mathbf{p}_1^T, \mathbf{p}_2^T)^T$ – is projected into the manifolds latent space using UKR's orthogonal projection $\hat{\mathbf{x}}(t) = \mathbf{g}(\mathbf{o}; \mathbf{X})$. Here, however, only the $p_1(t)$ is considered. **Step (3)**: Starting from the identified current position within the manipulation (i.e. $\hat{\mathbf{x}}(t)$), a certain number of latent parameters corresponding to the near future of the current position are determined. **Step (4)**: These future positions then are re-mapped into the observation space by means of the UKR mapping $\mathbf{f}(\mathbf{x}; \mathbf{X})$ yielding future ball positions $\mathbf{p}_1$ and $\mathbf{p}_2$ and – most importantly – the corresponding hand postures as represented in the manifold. **Step (5)**: Since the recorded finger movements constitute the *action* and the ball movements the corresponding *reaction* in the system, the actuation of the future hand postures determined in Step 4 can be expected to result in ball positions compatible with the manipulation as represented in the manifold. **Closing the loop**: after the finger movement in Step 5, the resulting change of the manipulation state – i.e. the displacement of the balls – is observed by the camera and the cycle restarts with Step 1.

used whereas the parameter-variance penalty is inactive ($\lambda_{pvar} = 0$) since no corresponding latent dimension is present.

With these settings, the Structured UKR manifolds have been trained for different inverse bandwidth parameters $\beta$ and evaluated with respect to their appropriateness for the specific task. To this end, the self-reconstructions $\hat{\mathbf{o}}_i = (\hat{\mathbf{\Theta}}^T, \hat{\mathbf{p}}_1^T, \hat{\mathbf{p}}_2^T)^T = \mathbf{f}(\mathbf{g}(\mathbf{o}_i))$ of the training observation vectors $\{\mathbf{o}_i\}$ have been computed as a decision basis.

Figure 5.18 shows the results for $\beta = 10, 20, 30$, and 40 from the perspective of the ball-1-trajectories: the points depict the $\mathbf{p}_1$-parts of the training data; their colour encode the root mean square error (RMSE) between the corresponding hand postures $\mathbf{\Theta}_i$ and $\hat{\mathbf{\Theta}}_i$. The red lines visualise the $\hat{\mathbf{p}}_1$ trajectory resulting from mapping the whole range of latent parameters $\mathbf{X}$ back to the observation space ($\hat{\mathbf{o}} = \mathbf{f}(\mathbf{X})$) giving an impression of the smoothness of the trained manifold.

Whereas the model for the smallest inverse kernel bandwidth parameter $\beta = 10$ (Figure 5.18(a)) consequentially yields the smoothest representation, it also results in the highest RMSEs for the reconstructed hand postures. Indeed, using this model for generating the ball swapping motion could not produce any ball swappings. By increasing $\beta$ (i.e. decreasing

the kernel bandwidth) to 20 (Figure 5.18(b)), the RMSEs decrease, but at the price of a less smooth representation as shown by the ball-1-trajectory. However, only a further increase of $\beta$ to 30 (Figure 5.18(c)) sufficiently improved the hand posture synthesis ability, especially in the bottom right part of the figure. This eventually led to a model that is able to reproduce the ball swapping in at least half of the open loop trials. A further increase of $\beta$ as shown in Figure 5.18(d) further decreases the RMSEs in most parts of the mapping, but also results in a more over-fitted solution. The model trained with $\beta = 30$ has therefore been chosen for the following process.

### 5.6.3. Feedback control using Structured UKR

The training of the Structured UKR manifold described in the previous section yields a representation of the underlying manipulation that provides on the one hand information about valid combinations of hand postures and ball position, and on the other hand knowledge about the chronological order of the data.

Exploiting these two main characteristics of the manifold, a closed-loop feedback control system can be implemented as shown in the schematic overview in Figure 5.19:

1. In the first step (Figure 5.19, box 1), the current ball positions $\mathbf{p}_1(t)$ and $\mathbf{p}_2(t)$ are extracted from the camera image as a description of the current manipulation state. These points are treated as partial observations of the manipulation: $\mathbf{o}^{\star}_{\mathbf{p}_1}(t) = (\star, \mathbf{p}_1(t)^T, \star)^T$ and $\mathbf{o}^{\star}_{\mathbf{p}_2}(t) = (\star, \star, \mathbf{p}_2(t)^T)^T$, respectively. Here, the $\star$ denotes unspecified values.

2. In the second step (Figure 5.19, box 2), the partial observations are mapped into the UKR latent space by means of the orthogonal projection $\hat{\mathbf{x}}(\mathbf{t}) = \mathbf{g}(\mathbf{o}; \mathbf{X})$ (see Section 2.1). Here, however, only the specified part of the data is used (e.g. $\mathbf{p}_1$ or $\mathbf{p}_2$, respectively). This partial observation projection then yields the latent parameter – and thus the point in time or phase of the manipulation – which best matches the observed ball-1-position. In addition, it also provides means to determine the corresponding full reconstruction $\mathbf{o}$ from the manifold representation using the UKR mapping $\mathbf{f}(\mathbf{x}; \mathbf{X})$ (see Section 2.1). Since balls 1 and 2 are identical, this reconstruction can be used to find the ball whose actual position best matches with its reconstruction. The ball position yielding the smallest reconstruction error defines the latent parameter $\hat{\mathbf{x}}(t)$ used in the following process.

3. For the manipulation control, however, the represented expected *future* development of the movement is of particular interest. Thus, in the third step (Figure 5.19, box 3), starting from the identified current latent position within the manipulation (i.e. $\hat{\mathbf{x}}(t)$), a list of $k$ successive 'future' latent parameters is generated. The latent parameters are sampled using $\Delta$ as fixed sample distance: $\mathbf{X}^{f}_{\hat{\mathbf{x}}(t)} = (\hat{\mathbf{x}}(t) + 1\Delta,\ \hat{\mathbf{x}}(t) + 2\Delta, \ldots,\ \hat{\mathbf{x}}(t) + k\Delta)$. This list constitutes the latent representation of a plan for a short-term feed-forward motion. The combination of $k$ and $\Delta$ specifies the coarseness/speed ("step size" $\Delta$) and the length/duration ($k \cdot \Delta$) of the feed-forward part of the control.

4. In step four (Figure 5.19, box 4), this plan is mapped back into the observation space by means of the UKR mapping $\mathbf{f}(\mathbf{x}; \mathbf{X})$. The resulting observations $\mathbf{O}^{f}_{\hat{\mathbf{x}}(t)} = (\hat{\mathbf{o}}^{1\Delta}_t,\ \hat{\mathbf{o}}^{2\Delta}_t,\ \ldots\ ,\ \hat{\mathbf{o}}^{k\Delta}_t)$ correspond to the expected actions (i.e. finger movements) and reactions (i.e. ball movements) from a proceeding manipulation.

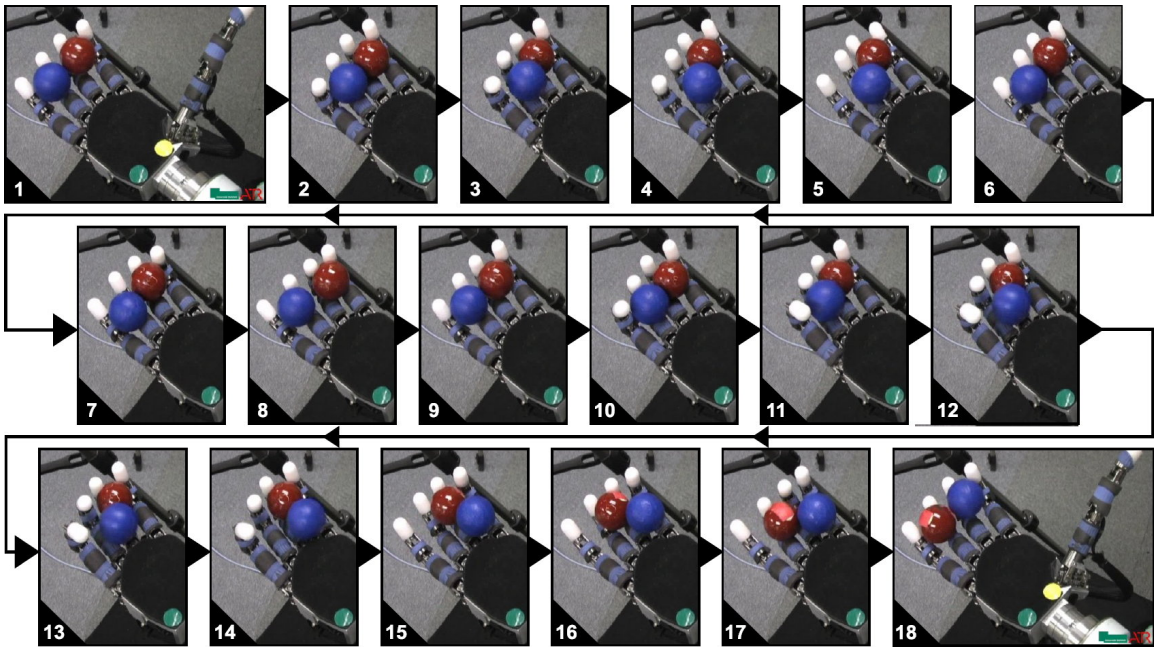



**Figure 5.20.:** Exemplary results of the UKR open loop control.

5. Since the changes of the ball positions are directly induced by the observed finger movements, however, the actuation of these actions can be expected to result in ball positions that are similar to the represented ones. In step five of the feedback control (Figure 5.19, box 5), these expected future hand postures are therefore actuated in their chronological order to perform the next steps of the desired manipulation.

After the finger movement, the resulting change of the manipulation state – i.e. the displacement of the balls – is observed by the camera and evaluated in order to adapt the hand motion and set up a new short-term feed-forward plan. In this way, the control loop is closed and restarts with step 1.

### 5.6.4. Experiments

The experiments focus on the qualitative evaluation of the differences in the behaviours of the UKR open loop and UKR closed-loop controllers.

The UKR *closed-loop* controller uses the control scheme presented in the previous section with the manifold described in Section 5.6.2. For the UKR *open loop* control, the same trained UKR manifold was used as a basis for the motion generation. In this case, no feedback about the current ball positions is used. Instead, the latent space is regularly sampled with a fixed step size $\Delta$ and the resulting latent value $x(k) = x_0 + k \cdot \Delta, (k = 1, 2, ..)$ is mapped back to observation space by means of $\mathbf{o}(k) = (\mathbf{\Theta}(k)^T, \mathbf{p}_1(k)^T, \mathbf{p}_2(k)^T)^T = \mathbf{f}(x(k))$. The corresponding hand joint angles $\mathbf{\Theta}(k)$ are actuated (while the ball positions $\mathbf{p}_1(k)$ and $\mathbf{p}_2(k)$ are ignored). The represented movement is therefore produced in a purely feed-forward manner. The speed of the movement can be varied with the size of $\Delta$.

Since both controls are based on exactly the same UKR representation of the underlying motion, the capability of performing the ball swapping is equally present in both methods. However, by incorporating the ball position feedback with the new control scheme

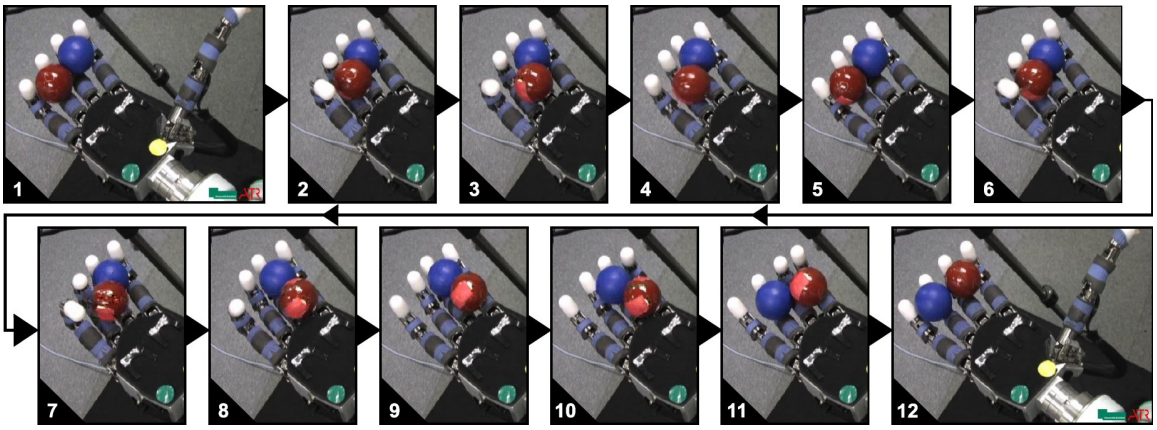



**Figure 5.21.:** Exemplary results of the UKR closed-loop control.

presented in the previous section, the robot gains the ability to adequately react to unforeseen events or control failures. The exemplary sequences shown in Figures 5.20 to 5.22 illustrate the different behaviours of open loop and closed-loop control for the same initial ball configuration.

Figure 5.20 shows intermediate hand postures from the open loop control. Here, Pictures 1 to 5 show an unsuccessful attempt to swap the balls. Indeed, since the open loop control cannot react to unachieved sub-goals of the control, the following movement continues as if the blue ball had been correctly moved between red ball and the palm (see Picture 14 for the targeted configuration). Pictures 6 to 9 show, how the middle finger and especially the index finger therefore try to bring the red ball to the position to which the blue ball returned. After this full cycle of the hand motion, both balls are in the same configuration as in Picture 1 without any swapping. In the following, Pictures 10 to 18 show a subsequent successful attempt to swap the balls with the open loop control.

Figure 5.21 depicts the same initial ball configuration as in Figure 5.20, but using the closed-loop control (on the basis of the same UKR representation). Corresponding to the open-loop control, Pictures 1 to 5 show an unsuccessful attempt to swap the balls. However, since the closed-loop control recognises that the targeted sub-goal has not been achieved (or rather: that the blue ball returned to its initial position), the adequate part of the control is repeated (Pictures 6 to 9) and the goal is eventually reached. Pictures 10 to 12 complete the ball swapping cycle.

Figure 5.22 shows a control scenario in which the blue ball is manually pushed back to the initial position in order to prevent the ball from swapping. In this sequence, three attempts to push the ball to the correct position can be seen in Pictures 1 to 4, 5 to 8, and 9 to 12. In each of these attempts, the controller recognises that the blue ball needs to be pushed in the direction of the palm and therefore actuates the little and ring finger. Since the ball is manually pushed back, the algorithm reapplies the adequate part of the control and tries the manipulation again.

Whereas the closed-loop manipulations (Figures 5.21 to 5.22) are not perfect and still yield errors in the ball swapping manipulation, the closed-loop control scheme better exploits the underlying UKR representation and realises a hand motion which is adapted to the current ball configuration. It therefore better reacts to unforeseen disturbances during the manipulation.

One interesting observation is that the repeated "trying" of the robot to accomplish the

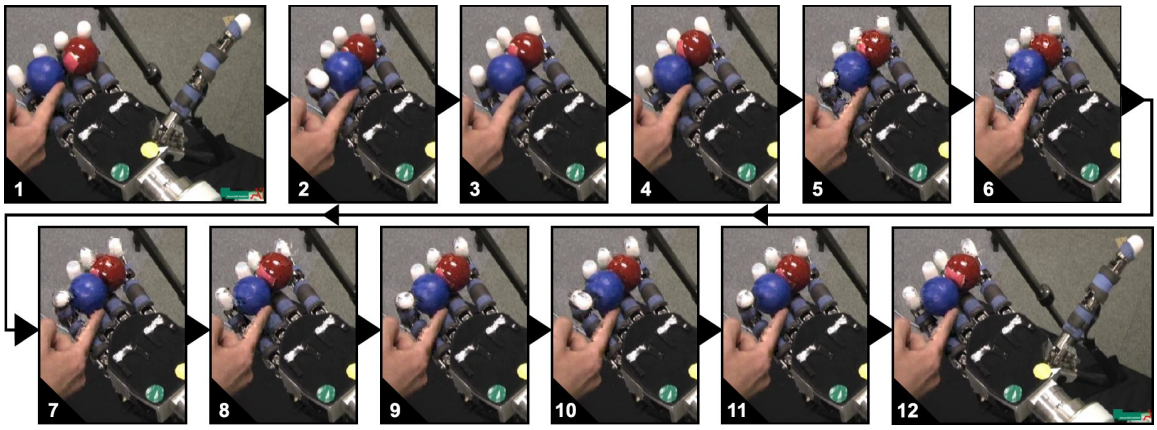



**Figure 5.22.:** Exemplary results of the UKR closed-loop control. Whereas the open loop control (see Figure 5.20) does not react to unforeseen situations, the closed-loop control adapts to the current manipulation context.

sub-goal of bringing one ball in a specific position gives the impression that the robot has a kind of awareness of the current situation yielding a very natural looking manipulation motion.

### 5.6.5. Discussion

In this section, a closed-loop controller scheme is described which operates on the predefined clear structure of Structured UKR manifolds - a modified version of Unsupervised Kernel Regression.

Using this controller scheme, the ball swapping task has been successfully implemented on a real 16 DOF robot hand, namely the Gifu III Hand, using the positions of the balls as sensory feedback.

In addition to the proof of the presented concept on a real robot, it could also be shown that the new control is able to better exploit the underlying motion representation in order to realise a more sophisticated control which adequately adapts to the current situation.

The resulting behaviour of repeatedly "trying" to accomplish a specific sub-goal gives the impression that the robot has a kind of *awareness* of the current situation yielding a very natural looking manipulation motion.

Particularly since the UKR closed-loop controller is based on exactly the same manifold as the open-loop version (which gives no impression of any comprehension of the robot about the task), this is a remarkable observation which serves to illustrate the gain of using the presented method.

# 6. Structured Manifolds for Motion Recognition

In the previous chapter, Structured UKR manifolds have been introduced as structured representations of motions for the purpose of their reproduction e.g. on a dextrous robot. The generation of such manifolds is based on observed training data from motion capturing and can be performed in a very robust, fast and easy manner. Due to the design of the generation procedure, the latent structure of the resulting manifolds is clear, known beforehand and, most importantly, unified for a wide range of applications.

Because of this unification, the idea for a general (motion) reproduction controller has been presented, which can be very easily transfered to a variety of domains of time series.

Indeed, since the manifold structure by design reflects the task-related structure of the represented motion, this reproduction controller reduces to very simple navigations within the manifold's latent space. In fact, the controller itself does not include any specification or knowledge of the underlying motion, but this knowledge exclusively and completely is encapsulated in the motion representation. Exploiting this knowledge captured in the representation is the key for the motion reproduction.

One important point concerning the manifold characteristics is that the manifolds are generated on the basis of motion capture data and therefore hold representations of motions *in the form of observations* and not as motor commands. In order to reproduce the underlying motions on a robot (which is the main goal of the representation), the corresponding motor commands are required and an adequate mapping from the observations is necessary.

Whereas the use of observable data is a small limitation on the one side, it also provides useful characteristics for a different kind of application. Most importantly, the manifolds consist of observations of recorded motions and the knowledge about the underlying motion that is represented in the manifold is directly available in terms of observable data.

These characteristics of Structured UKR manifolds cannot only be used in the direction described in the previous chapter, which is (a) initially choose a manifold corresponding to a desired motion (b) navigate through its latent space for retrieving targets from mapping the latent positions 'back' to observation space and (c) actuate the resulting intermediate postures.

Rather, due to the intrinsic observability of the data, the characteristics of Structured UKR manifolds moreover pave the way for the inverse direction. By providing a set of manifolds all representing different motions in the same manner, newly perceived observations of motions can be compared with the represented, previously learned *candidate motions* in order to evaluate to which of the candidates the observation is best compatible. In this manner, a system of Structured UKR manifolds can be used for the classification and recognition and even for the segmentation of observations according to the set of candidate motions, as will be described in the following sections.

The fact that the manifolds have a clear and unified structure by design, is again largely beneficial also in the case of recognition. Indeed, it is the fundamental characteristics which renders it possible to perform a meaningful comparison of the observations' compatibilities

with different manifolds.

Furthermore, since all candidates are represented in manifolds originally designed for the reproduction of motions, the resulting system is not only capable to recognise a set of candidate motions, but also inherently features the actuation of the same. Moreover, actuation and perception of one and the same motion corresponds to activity in the same part of the representation and the improvement of the underlying representation therefore affects both, the actuation and the perception abilities. The principle of such intrinsic combination of perception and action mechanisms is very useful in applications and, indeed, has not been invented in the field of robotics, but can rather be observed in nature. Here, neurophysiological findings, namely the mirror neuron theory (e.g. (Rizzolatti & Craighero, 2004; Falck-Ytter et al., 2006; Rizzolatti et al., 2009)), describes neurons in the premotor area of the brain (which is related to the actuation of motions) that show activity in both cases, when a subject performs an object-related action or observes the same action while performed by another subject.

In the following sections, two approaches of using Structured UKR manifolds for the recognition of represented patterns are presented. The basic methods have previously been published in (Steffen, Pardowitz, & Ritter, 2009a, 2009b; Steffen et al., 2010).

Other approaches exist in this field, which address the task of motion recognition or segmentation in fundamentally different ways. Especially in the context of robot task learning from human demonstration, the segmentation plays an important role. Arsenio (2004) applies hand-crafted rules to detect state transitions in video sequences. Segments are characterised through stable contact points between the objects recognised in the scene. More formalised models use Hidden-Markov-Models (HMMs) to segment walking or grasping actions from motion-capture data (Beth et al., 2003). Calinon and Billard (2004) perform unsupervised clustering using Vector Quantisation (VQ) to segment the basic actions (codes) for a discrete HMM. This method is refined in (Calinon et al., 2006) to Gaussian Mixture Models where each Gaussian represents a single segment of a task demonstration. This GMM is then fed into a continuous HMM for sequence learning. A taxonomy of action primitives is presented by Ehrenmann et al. (2003). These primitives of action (mainly concerned with grasping) are learned in a supervised way in (Zöllner et al., 2001). This allows to classify each frame of a task demonstration and to construct task segments from those classifications. These segments have been transformed into petri-nets for execution on a humanoid robot by Zöllner et al. (2004). A similar way is proposed by Bentivegna (2004) where a user demonstration is segmented based on the most likely primitives performed in each timestep. Nicolescu and Mataric (2001) apply a similar method using a winner-take-all selection of the most probable behaviour to segment a sequence of navigation tasks. Several methods try to avoid the segmentation problem: Iba et al. (2005) let the user define the segmentation with explicit verbal commands that directly guide the robot through a demonstration. Atkeson and Schaal (1997) and Suleiman et al. (2008) do not decompose a task demonstration at all but search for direct mapping functions between input and output trajectories.

## 6.1. The 'ABCD-data'

For the following part of the thesis, the domain of manipulation data, that has been used in the previous chapter, is left. Instead, the presentation of the recognition mechanism using

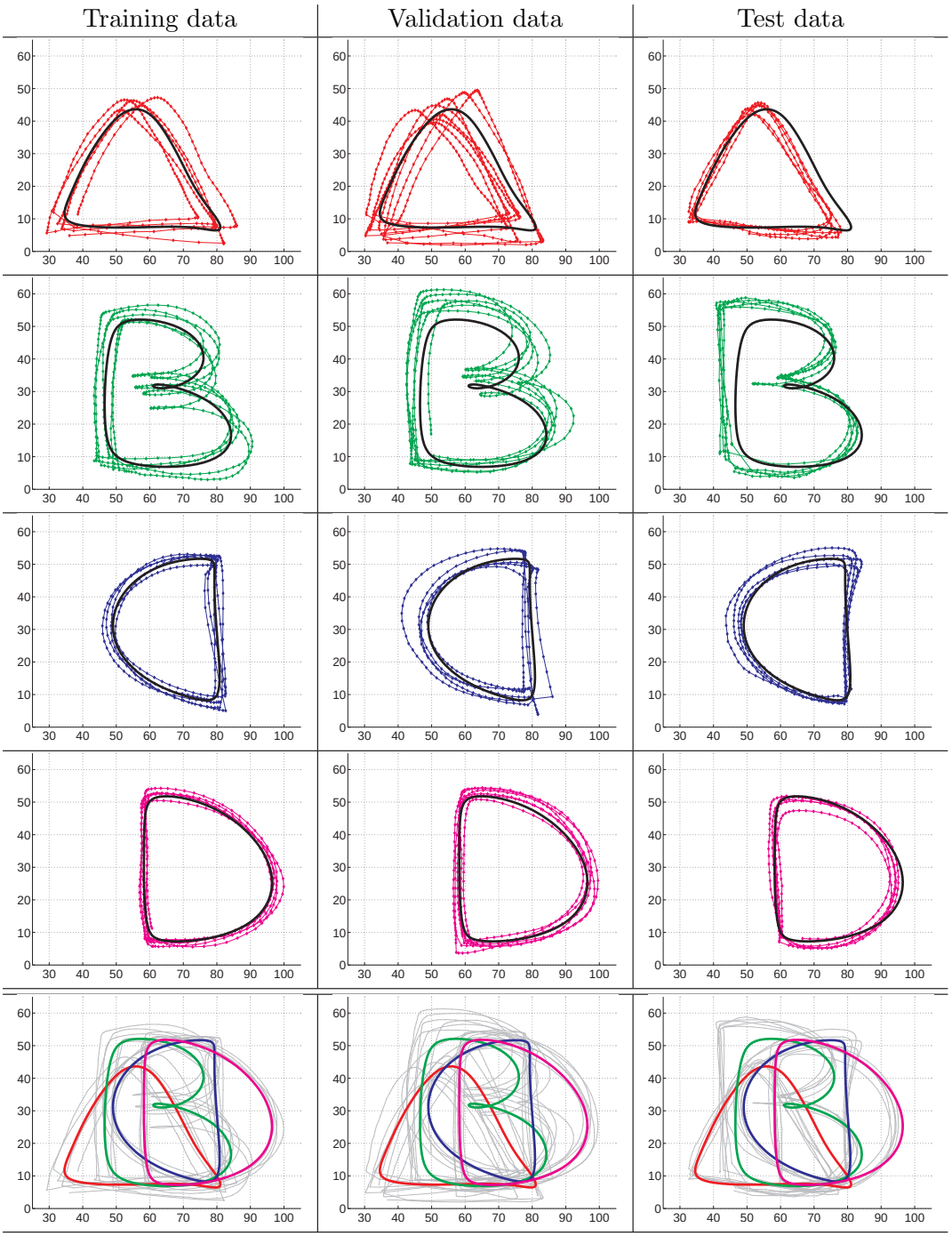



**Figure 6.1.: (left)** The ABCD-training data. The black bold lines visualise the UKR manifold representations resulting from the training with the depicted data. **(middle)** The validation data which has been used to optimise the model parameters of the UKR-only and the combined UKR/CLM approach. The black bold lines depict the same UKR representations as in the left column (trained on the basis of the training data). **(right)** The test data which has been used to evaluate both approaches. The black bold lines again depict the UKR representations. **The bottom row** depicts all UKR representations of the training data (coloured bold lines) along with all training, validation, or test data (gray lines), respectively.

Structured UKR manifolds concentrates on simpler and more descriptive data in order to facilitate the understanding of the method itself and the evaluation results, respectively.

The data – depicted in Figure 6.1 – consists of several two-dimensional trajectories of a hand that draws the letters 'A', 'B', 'C', and 'D' in the air, whereas the trajectory directions correspond to the natural hand writing directions. The 2D hand positions of each sample observation is extracted from the pictures of a monocular video camera which is placed on top of the scene, with orthogonal viewing direction onto the virtual 'drawing plane'. For the hand tracking, the ARToolkit[1] system has been used, whereas only the $x/y$-coordinates are considered since the $z$-component (height or distance to the camera, respectively) in the ARToolkit system is rather noisy and the orientations of the hand are irrelevant for the studied task. For each letter, three sets of sequences have been recorded[2]:

1. A training set, consisting of five trajectories, which has been used to train the UKR manifold representations (see Figure 6.1(left)).

2. A validation set, also consisting of five trajectories, which has been used to optimise the model parameters of the approaches (see Figure 6.1(middle)).

3. A test set, also consisting of five trajectories, which has been used to evaluate the approaches (see Figure 6.1(right)).

The UKR representations of the letters, that result from the training with the corresponding training set, are depicted as bold black lines in the top four rows of Figure 6.1, and all together as bold coloured lines in the bottom row.

## 6.2. Structured UKR for recognising movements

In order to rate the similarity of an observed motion sequence and a motion represented in a Structured UKR manifold, the two main characteristics of such manifolds can be exploited:

(a) the UKR training optimises a non-linear manifold model such that the underlying mapping $\mathbf{f}(\cdot)$ from the manifold's latent space into observation space yields a faithful representation of a set of given training observations. During this optimisation, in particular the self-reconstruction error of the training data is minimised. In consequence, the training observations lie on the manifold or very close to it afterwards. If the set of training observations is adequate to describe the underlying true manifold structure and the training is successful in capturing this structure, then the manifold constitutes a faithful generalisation of it. In this case, new observations corresponding to the same motion that is represented in the manifold also lie on the manifold or very close to it. The distance between observation and manifold, i.e. the self-reconstruction error, therefore can be used as a measure to rate if the evaluated observation is *compatible* with a manifold or not.

(b) the training of Structured UKR manifolds is modified in order to represent a set of series of observations whereas each series corresponds to one motion example. In particular, the training mechanism enforces the latent representation of the training

[1] http://artoolkit.sourceforge.net

[2] The data recordings of the training and validation data have been performed by Michael Pardowitz in collaboration with Nikolai Falke in the Neuroinformatics Group at the Bielefeld University.

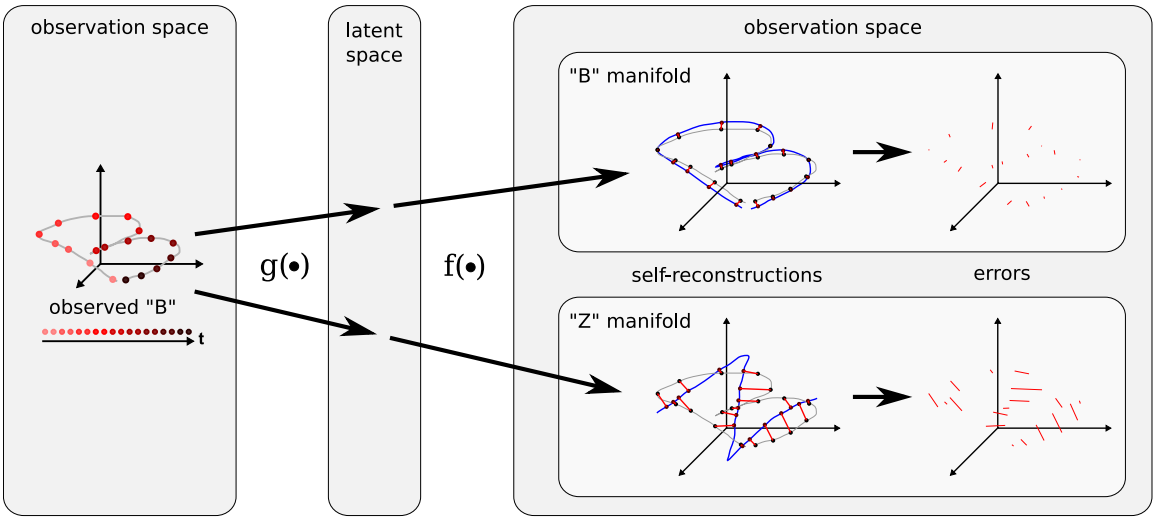



**Figure 6.2.:** Schematic visualisation of the reconstruction compatibility in observation space. The observed samples (red points, darker with development of time) are first projected into the manifold's latent space and then mapped back into original observation space: $\mathbf{f}(\mathbf{g}(\cdot))$ (see Section 2.1). Depending on the similarity of observed (grey line) and represented (blue line) movement, the resulting errors significantly differ: as shown, errors are very small when self-reconstructing observed "B" samples with a matching "B" manifold (upper pathway), and contrarily very high, when reconstructing the same samples with a non-matching manifold (e.g. "Z": lower pathway).

series to reflect the correct chronological order of these series. In consequence, the latent projections of the training data have the same chronological order as the training data. Again, if the set of training observations is adequate to describe the represented motion, then the manifold constitutes a faithful generalisation of it. In this case, the latent projections of a newly perceived series of observations corresponding to the represented motion also have the same chronological order as the observations. A measure that rates the degree of deviation of this correct chronological order therefore can be used as second measure to rate if the evaluated series of observations is *compatible* with a manifold or not.

Following these two general considerations, the recognition approach takes the inverse direction to the motion production described in the previous chapter: instead of projecting latent trajectories into the observation space in order to determine a sequence of intermediate targets for a subsequent actuation, such sequences are now observed and are treated as input data for a recognition/segmentation problem. By projecting them from observation space into the manifold's latent space by means of the approximation function $\mathbf{g}(\cdot)$ (see Section 2.1) and back to observation space using $\mathbf{f}(\mathbf{g}(\cdot))$, the features described above can be denoted in the domain of the manifold. These features are not directly connected to the domain of the represented data, but rather express the degree of compatibility (or similarity) of the observations and the considered candidate manifold.

In this context, the observations are described by a chronologically ordered series of recorded data samples $\mathbf{Y}^o = \{\mathbf{y}^o_1, \ldots, \mathbf{y}^o_M\}$ which are to be assigned to one of the underlying candidate motions. The corresponding latent projections $\hat{\mathbf{x}}^o_t = \mathbf{g}(\mathbf{y}^o_t)$ of these samples together with their self-reconstructions into observation space $\hat{\mathbf{y}}^o_t = \mathbf{f}(\hat{\mathbf{x}}^o_t)$ then constitute the basis for the following two recognition features:

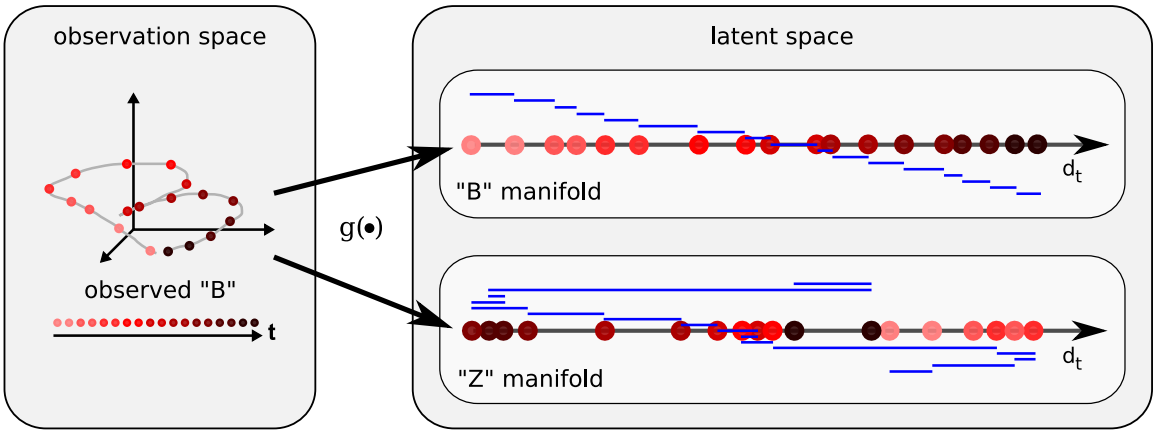



**Figure 6.3.:** Schematic visualisation of the history compatibility in latent space. Corresponding to Figure 6.2, "B" samples are projected into the latent space of candidate manifolds (red points, darker with development of time); if observed and represented movements correspond to each other (top row, projecting "B" samples onto "B" manifold), the latent projection order matches the observation order and the sum of successor distances is minimal (blue lines). Otherwise (e.g. bottom row, projecting "B" samples onto "Z" manifold), the successor distances significantly differ from the matching case (see the overall length of blue lines in top and bottom schema).

(a) The compatibility of single observations $\mathbf{y}_t^o$ with the manifold is reflected in its distance to the UKR self-reconstruction: $\| \Delta(t) \| = \| \mathbf{y}_t^o - \hat{\mathbf{y}}_t^o \|$. Depending on the degree of similarity of observed and represented data, this distance varies significantly, as schematically visualised in Figure 6.2. On this basis, the *self-reconstruction compatibility* can be denoted as:

$$C_{rec}(t; \mathbf{Y}) = -1 + 2 \cdot \exp\left(-\Delta^T \mathbf{\Sigma}^{-1} \Delta\right) \tag{6.1}$$

where $\mathbf{\Sigma}(\mathbf{Y})^{-1} = \mathrm{diag}(\frac{1}{\mathrm{var}(\mathbf{y}_{\cdot,1})}, \ldots, \frac{1}{\mathrm{var}(\mathbf{y}_{\cdot,d})})$ is used to weigh the dimensions according to the corresponding variance in the UKR training data $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$.

In this formulation, $C_{rec}(t; \mathbf{Y})$ can take values in the interval $[-1; +1]$ whereas $-1$ reflects worst and $+1$ best compatibility with the underlying UKR manifold.

(b) The chronological compatibility of observations in terms of the considered candidate manifold is reflected in the order of their latent projections according to the latent time dimension.

The basis for an adequate measure constitutes the following observation (schematically visualised in Figure 6.3): if the evaluated candidate manifold matches the series of processed observations, the latent projections of successive observations are also correctly ordered and have a small positive directed distance to each other (corresponding to the latent time dimension). In Figure 6.3, this case is depicted in the upper pathway: when projecting an observed '$B$' trajectory into the latent space of a '$B$' manifold, the projections of later observations have a higher latent time value than the projections of prior observations whereas the distance between two successive projections is small. In contrast, the projection of series of observations into the latent space of a non-matching manifold yields a different picture. The lower pathway in Figure 6.3 visualises such a situation for an observed '$B$' trajectory that is projected into the latent space of a '$Z$' manifold. Whereas some pairs of successive

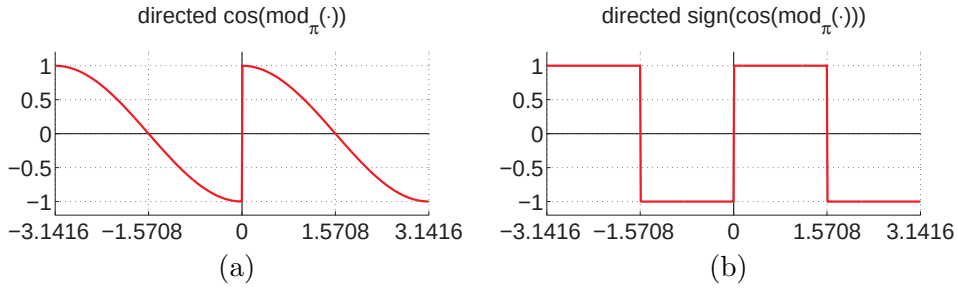



**Figure 6.4.: (a)** By computing the cosine of the modulo of $\pi$, the distance compatibility measure differentiates the directions of the deviations. Correct order yield positive, incorrect orders negative values. **(b)** By only considering the sign of (b), the measure becomes a decision function that expresses if the order is correct ($= 1$) or not ($= -1$). Whereas it does not distinguish different levels of latent proximity, it is also not sensitive to different sampling rates.

projections yield the same characteristics as the matching case described above, there are also pairs with a negative or very large positive distances. These 'jumps' in the series of successive latent projections are symptomatic for non-matching manifolds such as the missing of these jumps is symptomatic for matching manifolds. A further supporting characteristics can be observed by comparing the figures 6.2 and 6.3: even if the order of the projections matches with the generally non-matching manifold, then the self-reconstruction errors of the corresponding observations are often very large.

These considerations can be used as a basis for a measure that rates the local compatibility of an observation $\mathbf{y}_t^o$. It consists of a balanced combination of the self-reconstruction compatibility $C_{rec}(t)$ of the evaluated observation and the chronological compatibility of its latent projection $\hat{\mathbf{x}}_t^o$ with the prior projection $\hat{\mathbf{x}}_{t-1}^o$:

$$c_{local}(t) = \frac{1}{2}\left(C_{rec}(t) + C_{ord}(t)\right). \tag{6.2}$$

where $C_{rec}(t)$ is the self-reconstruction compatibility defined in (a) and $C_{ord}(t)$ is the order compatibility, that works on the latent time dimension $d_t$:

$$C_{ord}(t) = H_{\circlearrowleft}^{\pi}\left((\hat{\mathbf{x}}_{t-1}^o)_{d_t} - (\hat{\mathbf{x}}_t^o)_{d_t}\right) \tag{6.3}$$

$$\text{with e.g. } H_{\circlearrowleft}^{\pi}(x) = \cos\left(\text{mod}_{\pi}(x)\right). \tag{6.4}$$

The function $H_{\circlearrowleft}^{\pi}(\cdot)$ rates the compatibility of directed distances in the latent time dimension $d_t$ which is periodic in $\pi$. To this end, different possibilities are viable, depending on the desired behaviour of the measure. Figure 6.4 exemplarily depicts two of such functions that rate the compatibilities as $+1$ for maximally compatible and $-1$ for maximally incompatible distances:

Figure 6.4(a) depicts the cosine of the modulo of $\pi$ which constitutes a measure that yields values close to 1 for small positive deviations, decreases to 0 for a deviation of $\frac{\pi}{2}$ (which is the maximal distance in a dimension with period $\pi$) and further decreases towards $-1$ for deviations close to, but below $\pi$. Due to the modulo-$\pi$-operator, small negative deviations are also treated as maximally incompatible and therefore causes the 'directedness' of the rating function.

Figure 6.4(b) illustrates the sign function applied on the rating function depicted in Figure 6.4(a). In consequence, the resulting graph corresponds rather to a decision function than to a rating. Since no distinction between different degrees of proximity are made, this function is less sensitive to different sampling rates of the observations, but neglects different levels of compatibility.

In order to express the compatibility of the time course of observations prior to the evaluated observation $\mathbf{y}_t^o$, the local compatibilities of the corresponding history of length $H$ is taken into account. The *history compatibility measure* therefore is denoted as average local compatibility of prior observations. A discount factor $\gamma \in [0; 1]$ rates previous observations that are chronologically closer to the evaluated observation as more important than those farther back in the history.

For a history of length $H$ of the evaluated observation $\mathbf{y}_t^o$ , the *history compatibility* can be denoted as:

$$C_{hist}(t, H) = \frac{\sum_{h=1}^{H} \gamma^h c_{local}(t-h)}{\sum_{h=1}^{H} \gamma^h} \tag{6.5}$$

Corresponding to $C_{rec}$, also $C_{hist}$ can take values in $[-1; +1]$ whereas $-1$ reflects worst and $+1$ best compatibility with the underlying UKR manifold.

The combination of (a) and (b) to one overall compatibility measure using $\lambda \in [0; 1]$ as weighting factor yields:

$$C = \lambda C_{rec} + (1 - \lambda) C_{hist} \qquad \in [-1; +1]. \tag{6.6}$$

$C$ provides a measure for the compatibility of the considered candidate manifold and the evaluated observation together with its history. In other words, $C$ realises a measure to quantify the appropriateness of a candidate manifold to reproduce the observation together with the observed history.

The classification of the observation to one of several candidate classes then is performed by a winner-take-all mechanism that works on the results of all UKR manifolds corresponding to the set of candidate classes. The classification system therefore chooses the class with the maximal compatibility to the observed motion.

### 6.2.1. Evaluation

The method is generally applicable to all sequence data that is represented in Structured UKR manifolds. The following evaluation is performed on the 'ABCD dataset' described in Section 6.1 since the letter trajectories yield intuitive interpretations and the corresponding results are particularly illustrative.

For the evaluation of the presented method, four Structured UKR manifolds have been trained – each on the basis of the training set of one of the letters 'A', 'B', 'C', or 'D'. The represented trajectories are shown in Figure 6.1.

In order to obtain adequate values for the parameters $\gamma$ and $\lambda$ (see Equations (6.5) and (6.6)), the method has been analysed for different parameter choices on the validation data set depicted in Figure 6.1(middle). In this process, values of $\gamma = 0.9$ and $\lambda = 0.3$ have been identified as a good choice for the following evaluation of the method on the test data set shown in Figure 6.1(right).

The evaluation of the compatibility works on the basis of single observations (together with their histories) and thus, the classification of trajectories does not require a fixed

amount of observations or recordings with a fixed sample rate. In this sense, a classification of a trajectory as a whole is not performed directly but emerges from the classification of succeeding observations to the same class.

The pointwise assignment of a class is based on the corresponding compatibility values of all candidate motions. A winner-take-all mechanism chooses the candidate with the highest compatibility value as final classification result.

In the following sections, three different experiments are described and evaluated.

The first one focusses on the general classification abilities in a 'clean' test scenario with trajectories that correspond to single whole letters, i.e. consists of a series of observations describing a single complete letter.

In order to evaluate the method in the context of segmentation, the second experiment evaluates the classification performance on trajectories that consist of two concatenated letter sequences. Since the classifications are basically pointwise with an additional consideration of a history part, the main focus in this experiment is drawn on the letter transition point within the trajectory.

The third experiment presents and evaluates another application possibility of the system where only middle parts of letter trajectories are observed. Two of such parts are concatenated in order to evaluate the classification and segmentation abilities, again with a major focus on the transitions between the two letters.

Whereas the three experiments described above evaluate the performance on the recognition of test sequences each corresponding to one of the known candidate classes, the subsequent section addresses the robustness of these classifications and the rejection of observations that do not correspond to any of the known classes.

In the following discussion, the advantages and drawbacks of the method are summarised.

### 6.2.2. $1^{st}$ Experiment: Classification of whole-letter-trajectories

In the first experiment, the method for the classification of trajectories is evaluated corresponding to single and complete letter trajectories for 'A', 'B', 'C', or 'D', respectively.

Figure 6.5 exemplarily visualises a complete evaluation run for the sequence '$B_4$' with all corresponding versions of different sampling rates and history lengths. Each sub-figure depicts the detailed pointwise classification results for one of these versions. The columns hold the different sampling rates $N^{th}$ (see below), whereas the rows correspond to the evaluated history lengths $H$.

Table 6.1 provides an overview of the results of all evaluations. In order to demonstrate that the method is able to cope with different sampling rates and data counts, all sequences are evaluated using *every* point of the test trajectory ($N^{th} = 1$), only *every second* ($N^{th} = 2$) or only *every third* ($N^{th} = 3$). In addition, the case $N^{th} = 4$ is listed in Appendix B.1.1. To get a better grasp on the distribution of the values in the table, high classification rates are marked with green background and low values with red. Displayed are the percentages of true positive classifications of single trajectory points.

Most of the results in Table 6.1 are very promising: 171 of all 237 evaluations (72.15%) yield classification rates of equal or above 85%. In contrast to that, there are also letters that yield classification results significantly below the average. Especially the test letters '$C_3$' and $C_5$ do not yield classification rates above 70% for all sampling rates and history lengths.

In general, when using the short history length with the fully sampled trajectories (i.e. $N^{th} = 1, H = 5$), the results stay significantly below those with longer history for all test

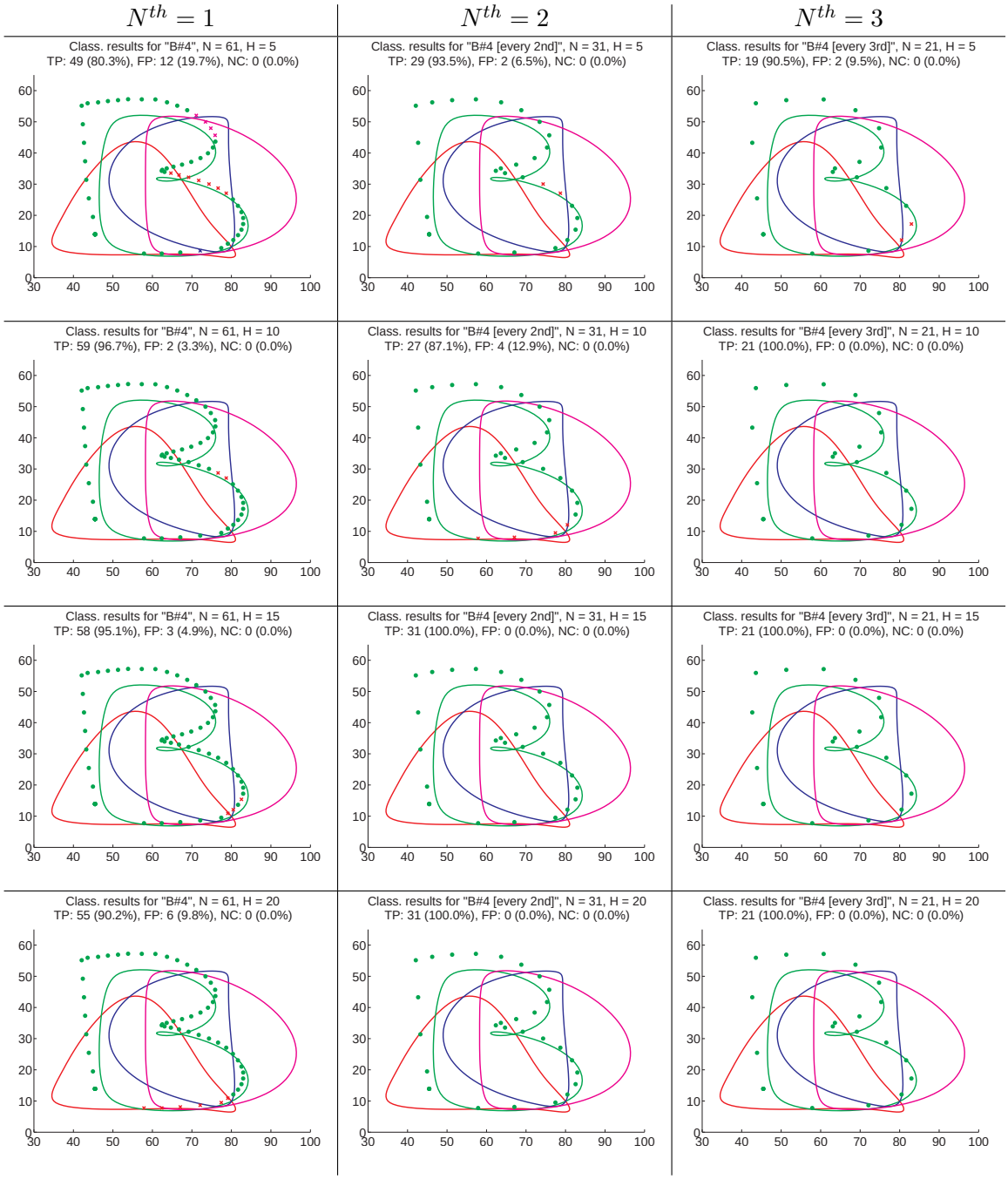



**Figure 6.5.:** Classification details for trajectories of single letters 'B'. Solid lines represent the four considered candidate UKR manifolds: 'A'(red), 'B'(green), 'C'(blue), and 'D'(magenta). Points and crosses denote the observations of the test trajectory whereas coloured *points* depict true positive ('TP') and coloured *crosses* false positive ('FP') classifications. Here, the colours encode the determined class label. The headlines of the sub-figures denote the name of the dataset (here: "$B_4$"), the size $N$ of the dataset, the history length $H$, the amount of true positive (TP) and false positive (FP) classifications and the amount of rejected points (NC: not classified). Depicted are the classification results for different history lengths $H = 5, 10, 15$ and 20 (rows 1-4, respectively) and different sampling rates: every point, every second, and every third (columns 1-3, respectively).

| | $N^{th}=1$ | | | | $N^{th}=2$ | | | | $N^{th}=3$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1$ | 84.3 | 95.7 | 100.0 | 100.0 | 94.3 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| $A_2$ | 84.1 | 92.1 | 98.4 | 100.0 | 93.8 | 100.0 | 100.0 | 100.0 | 95.2 | 100.0 | 100.0 | 100.0 |
| $A_3$ | 91.7 | 98.3 | 100.0 | 100.0 | 96.7 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| $A_4$ | 80.6 | 91.9 | 93.5 | 96.8 | 96.8 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| $A_5$ | 76.0 | 86.0 | 94.0 | 100.0 | 84.0 | 100.0 | 100.0 | 100.0 | 94.1 | 100.0 | 100.0 | n/a |
| $B_1$ | 56.4 | 76.9 | 83.3 | 83.3 | 84.6 | 94.9 | 94.9 | 94.9 | 80.8 | 80.8 | 92.3 | 92.3 |
| $B_2$ | 68.2 | 80.7 | 89.8 | 88.6 | 86.4 | 90.9 | 90.9 | 93.2 | 93.3 | 93.3 | 96.7 | 96.7 |
| $B_3$ | 69.0 | 87.3 | 80.3 | 83.1 | 83.3 | 80.6 | 77.8 | 88.9 | 91.7 | 91.7 | 91.7 | 91.7 |
| $B_4$ | 80.3 | 96.7 | 95.1 | 90.2 | 93.5 | 87.1 | 100.0 | 100.0 | 90.5 | 100.0 | 100.0 | 100.0 |
| $B_5$ | 71.8 | 79.5 | 80.8 | 82.1 | 74.4 | 84.6 | 87.2 | 92.3 | 80.8 | 84.6 | 92.3 | 92.3 |
| $C_1$ | 84.5 | 91.4 | 91.4 | 91.4 | 89.7 | 89.7 | 89.7 | 89.7 | 85.0 | 85.0 | 85.0 | 85.0 |
| $C_2$ | 62.9 | 74.3 | 77.1 | 80.0 | 71.4 | 74.3 | 85.7 | 85.7 | 75.0 | 87.5 | 87.5 | 87.5 |
| $C_3$ | 53.4 | 65.8 | 63.0 | 67.1 | 64.9 | 59.5 | 67.6 | 64.9 | 60.0 | 68.0 | 60.0 | 60.0 |
| $C_4$ | 80.0 | 81.7 | 85.0 | 93.3 | 80.0 | 93.3 | 93.3 | 93.3 | 75.0 | 85.0 | 85.0 | 85.0 |
| $C_5$ | 63.3 | 63.3 | 60.0 | 61.7 | 56.7 | 60.0 | 63.3 | 56.7 | 55.0 | 55.0 | 55.0 | 55.0 |
| $D_1$ | 95.6 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| $D_2$ | 92.1 | 96.8 | 96.8 | 96.8 | 93.8 | 93.8 | 93.8 | 93.8 | 95.2 | 95.2 | 95.2 | 95.2 |
| $D_3$ | 96.8 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| $D_4$ | 83.6 | 94.5 | 100.0 | 100.0 | 96.4 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | n/a |
| $D_5$ | 94.5 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | n/a |

**Table 6.1.:** Overview of the classification results for the evaluations of whole single letter sequences. The sequence name (first column) indicates the evaluated letter ('A', 'B', 'C', and 'D') and the corresponding test trajectory number ($1-5$ for each letter). The sequences are evaluated for different sampling rates, either considering *every* observation point ($N^{th}=1$), *every second* ($N^{th}=2$), or only *every third*, i.e. $N^{th}=3$ ($N^{th}=4$, see Table B.1 in Appendix B.1.1). For each sampling rate, evaluations are performed for four different history lengths, i.e. $H=5,10,15$, and 20. The listed values constitute the percentage of true positive point classifications for the processed sequence. The background colours of the table cells encode the underlying classification result, whereas green corresponds to good results and red to very poor ones. Cells holding the text 'n/a' on white background indicate that no evaluation has been done for the corresponding sequence because the history length $H$ is higher than the available point count of the evaluated sequence. Since every test sequence is evaluated separately and the evaluation procedure is deterministic, only one evaluation run has been performed for every test sequence.

sequences. This result indicates that the short history is not able to capture the trajectory structure in a sufficient manner and the compatibility measure acts in a too local context. For a sparser sampling (i.e. $N^{th}>1$), the history of length $H=5$ covers a relatively larger part of the overall motion and therefore yields better results.

The letter '$C_5$' yields the worst results in the evaluations listed in Table 6.1. Figure 6.6 illustrates the corresponding single classification results for the history length $H=15$ and the sampling rates (a) $N^{th}=1$, (b) $N^{th}=2$, and (c) $N^{th}=3$. The figure reveals that the first part of the '$C_5$'-trajectory (the left 'bow') is mostly confused with the '$B$' representation. By focussing on the relation between the '$C_5$' test sequence and the representations of the '$B$' and '$C$' candidates, and considering the effect of the compatibility measures, this result however is logically consistent: since the absolute self-reconstruction errors of the observations are directly considered as an indicator for compatibility, and the absolute positions of the observations therefore are taken into account, the first part of the '$C_2$'-trajectory in fact is more similar to '$B$' than it is to '$C$'. In addition, the chronological order of the latent space projections in both cases yield positive support, and does therefore

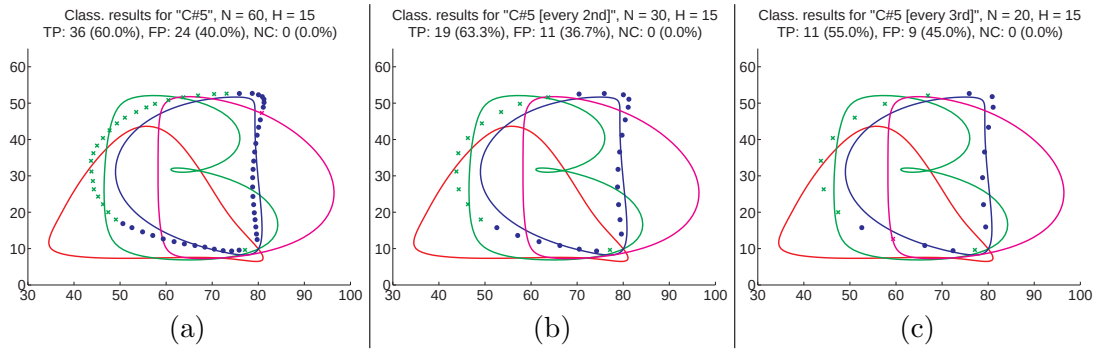



**Figure 6.6.:** Classification details for trajectories of single letters '$C_5$' for the history length $H = 15$ and sampling rates (a) $N^{th} = 1$, (b) $N^{th} = 2$, and (c) $N^{th} = 3$. General descriptions of the plot colours and data can be found in Figure 6.5.

not influence the classification significantly.

### 6.2.3. $2^{nd}$ Experiment: Segmentation of concatenated whole letter trajectories

The second experiment addresses the *segmentation* of series of observations according to the set of candidate classes. As before, the segmentation of trajectories is independent of the number of observations since each observation is processed separately. In principle, the classification of single points therefore is not affected by the concatenation of multiple letters in one trajectory and the segmentation can be performed on the basis of the single point classifications. Due to the inclusion of a history, however, especially the behaviour of the single point classifications at transition points between two letters require further inspection.

Indeed, Figure 6.7 shows that the classification results for concatenated letters do not differ largely from those of single letters (see Table 6.1): combinations of letters that yield good classification results in the single letter case as shown in Figure 6.7(a) and (b) also yield good results in the concatenated case. Consistently, combinations of letters that yield poor results in the single letter case also yield poor results when they are concatenated (see

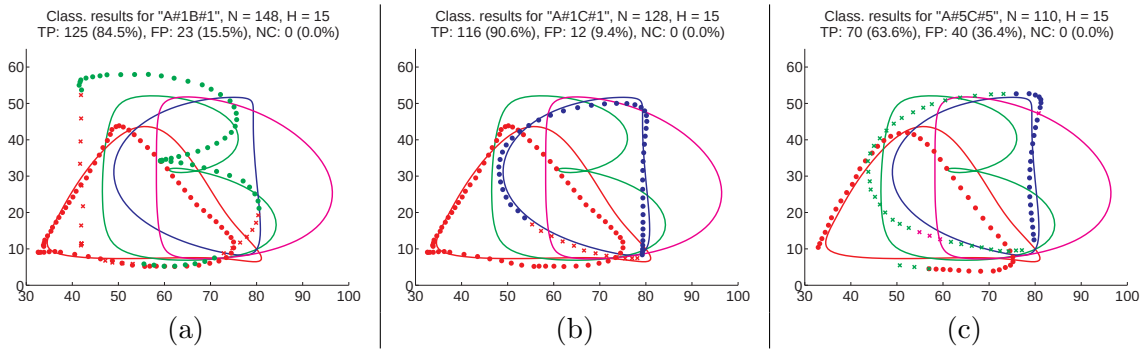



**Figure 6.7.:** Evaluation details for the combined sequences (a) '$A_1B_1$', (b) '$A_1C_1$', and '$A_5C_5$', each for the history length $H = 15$ and full sampling rate ($N^{th} = 1$). Since '$A_1$', '$B_1$', and '$C_1$' yield good results in the single letter evaluation (see Table 6.1), their combinations yield also good results. '$C_5$' evaluated individually, however, yields very poor results. Combinations with '$C_5$' as exemplarily depicted in (c) therefore also yield poor results for the '$C_5$' part. General descriptions of the plot colours and data can be found in Figure 6.5.

| | $N^{th} = 1$ | | | | $N^{th} = 2$ | | | | $N^{th} = 3$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1B_1$ | 67.6 | 80.4 | 84.5 | 82.4 | 85.1 | 91.9 | 93.2 | 91.9 | 86.0 | 86.0 | 92.0 | 86.0 |
| $A_1C_1$ | 84.4 | 90.6 | 90.6 | 85.2 | 89.1 | 85.9 | 78.1 | 73.4 | 90.9 | 79.5 | 72.7 | 56.8 |
| $A_1D_1$ | 87.0 | 93.5 | 92.8 | 89.1 | 92.8 | 89.9 | 88.4 | 78.3 | 93.6 | 89.4 | 78.7 | 78.7 |
| $A_2B_2$ | 74.2 | 82.1 | 86.1 | 86.8 | 85.5 | 86.8 | 88.2 | 89.5 | 86.3 | 88.2 | 88.2 | 86.3 |
| $A_2C_2$ | 72.9 | 80.5 | 83.5 | 80.5 | 80.6 | 79.1 | 77.6 | 71.6 | 82.2 | 77.8 | 71.1 | 64.4 |
| $A_2D_2$ | 86.5 | 90.5 | 92.9 | 92.1 | 90.6 | 93.8 | 93.8 | 81.3 | 90.5 | 92.9 | 78.6 | 73.8 |
| $A_3B_3$ | 78.6 | 87.8 | 86.3 | 88.5 | 84.8 | 87.9 | 83.3 | 86.4 | 90.9 | 90.9 | 86.4 | 81.8 |
| $A_3C_3$ | 70.7 | 78.9 | 75.9 | 74.4 | 77.6 | 73.1 | 73.1 | 71.6 | 75.6 | 75.6 | 66.7 | 66.7 |
| $A_3D_3$ | 91.9 | 94.3 | 94.3 | 92.7 | 93.5 | 90.3 | 93.5 | 87.1 | 92.7 | 92.7 | 82.9 | 73.2 |
| $A_4B_4$ | 80.5 | 94.3 | 94.3 | 93.5 | 95.2 | 93.5 | 96.8 | 88.7 | 95.2 | 97.6 | 88.1 | 85.7 |
| $A_4C_4$ | 79.5 | 82.8 | 83.6 | 82.8 | 83.6 | 85.2 | 78.7 | 73.8 | 85.4 | 80.5 | 70.7 | 63.4 |
| $A_4D_4$ | 77.8 | 88.0 | 88.0 | 89.7 | 91.5 | 93.2 | 88.1 | 81.4 | 92.5 | 92.5 | 80.0 | 75.0 |
| $A_5B_5$ | 73.4 | 82.8 | 86.7 | 90.6 | 78.1 | 90.6 | 90.6 | 89.1 | 86.0 | 88.4 | 88.4 | 88.4 |
| $A_5C_5$ | 65.5 | 65.5 | 63.6 | 67.3 | 61.8 | 67.3 | 69.1 | 63.6 | 64.9 | 67.6 | 67.6 | 51.4 |
| $A_5D_5$ | 81.9 | 89.5 | 95.2 | 98.1 | 88.7 | 98.1 | 92.5 | 81.1 | 94.4 | 94.4 | 80.6 | 72.2 |

**Table 6.2.:** Overview of the classification results for the evaluations of two concatenated whole letter trajectories. Additional evaluations can be found in Tables B.2 to B.4 in Appendix B.1.2. General descriptions of the table structure and colours can be found in Table 6.1.

Figure 6.7(c)).

Most of the single point classifications indeed correspond to the single whole letter case. Only at the transitions from one letter to the next, a *momentum-like effect* can be observed: for the corresponding trajectory part, the evaluated observations already belong to the beginning of a new letter, whereas parts of the considered history represent observations of the end of the previous letter. The inclusion of the history is therefore rather distorting for the transitions between trajectories of two letters. In Figures 6.7(a-b), this effect is nicely visible. Both test sequences start with the '$A_1$' trajectory. At the transition point to the next letter, i.e. '$B_1$' or '$C_1$' respectively, the first few observations of the new letter are wrongly classified as '$A$', since parts of the considered history still explain the former '$A_1$' trajectory.

Table 6.2 lists the evaluation results for combinations of the letter '$A$' test sequences with the '$B$', '$C$', and '$D$' test sequences. The names of the sequences listed in the table indicate the order of the concatenated whole letter sequences, e.g. '$A_1B_1$' is a sequence starting with the whole test sequence '$A_1$' followed by the whole test sequence '$B_1$'. Due to the high number of possible combinations of the 20 test sequences ($N = 20^2 = 400$), only a subset has been evaluated. Some additional evaluations can be found in Appendix B.1.2.

Logically consistent with the argument given above, the combinations of letters that yield poor classification results when evaluated individually (i.e. sequences '$C_3$' and especially '$C_5$', see Table 6.1), yield worse classification results than combinations with good individual classifications.

The general loss of classification performance that is observable in comparison to the single letter evaluations in Table 6.1 results from the momentum-like effect described above. Since the effect is mainly caused by the consideration of history points that do not correspond to the actual evaluated letter, and a longer history increases the number of observations for which such situation occurs, the momentum-like effect decreases with smaller history lengths. On the other hand, as already shown in the evaluation of whole single letters (see Table 6.1), the classification performance generally benefits from longer histories, and especially the case of the fully sampled trajectories with the short history length, i.e.

| | $N^{th}=1$ | | | | $N^{th}=2$ | | | | $N^{th}=3$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1^m B_1^m$ | 71.8 | 89.7 | 93.6 | 88.5 | 97.5 | 95.0 | 87.5 | 72.5 | 92.6 | 88.9 | 85.2 | 70.4 |
| $A_1^m C_1^m$ | 80.9 | 88.2 | 83.8 | 79.4 | 85.7 | 80.0 | 65.7 | 57.1 | 79.2 | 62.5 | 50.0 | 45.8 |
| $A_1^m D_1^m$ | 90.3 | 95.8 | 90.3 | 86.1 | 97.3 | 89.2 | 75.7 | 73.0 | 88.0 | 72.0 | 68.0 | 64.0 |
| $A_2^m B_2^m$ | 82.1 | 91.0 | 94.9 | 91.0 | 100.0 | 97.5 | 95.0 | 90.0 | 96.2 | 92.3 | 88.5 | 73.1 |
| $A_2^m C_2^m$ | 64.3 | 72.9 | 74.3 | 74.3 | 72.2 | 72.2 | 69.4 | 58.3 | 75.0 | 75.0 | 62.5 | 54.2 |
| $A_2^m D_2^m$ | 89.4 | 97.0 | 90.9 | 86.4 | 100.0 | 88.2 | 76.5 | 67.6 | 90.9 | 77.3 | 68.2 | 63.6 |
| $A_3^m B_3^m$ | 86.8 | 89.7 | 88.2 | 88.2 | 91.4 | 91.4 | 94.3 | 88.6 | 79.2 | 95.8 | 87.5 | 75.0 |
| $A_3^m C_3^m$ | 61.4 | 65.7 | 54.3 | 57.1 | 66.7 | 58.3 | 61.1 | 58.3 | 54.2 | 58.3 | 54.2 | 50.0 |
| $A_3^m D_3^m$ | 96.9 | 92.2 | 87.5 | 81.3 | 93.9 | 84.8 | 69.7 | 66.7 | 86.4 | 68.2 | 63.6 | 63.6 |
| $A_4^m B_4^m$ | 77.3 | 97.0 | 97.0 | 93.9 | 94.1 | 97.1 | 91.2 | 85.3 | 95.5 | 86.4 | 68.2 | 63.6 |
| $A_4^m C_4^m$ | 81.3 | 81.3 | 76.6 | 82.8 | 87.9 | 84.8 | 72.7 | 60.6 | 81.8 | 72.7 | 54.5 | 68.2 |
| $A_4^m D_4^m$ | 80.6 | 93.5 | 88.7 | 77.4 | 93.8 | 84.4 | 75.0 | 71.9 | 90.5 | 81.0 | 71.4 | 71.4 |
| $A_5^m B_5^m$ | 82.4 | 91.2 | 95.6 | 91.2 | 85.7 | 91.4 | 100.0 | 82.9 | 91.3 | 95.7 | 78.3 | 60.9 |
| $A_5^m C_5^m$ | 60.3 | 63.8 | 60.3 | 62.1 | 60.0 | 63.3 | 60.0 | 50.0 | 60.0 | 65.0 | 60.0 | 60.0 |
| $A_5^m D_5^m$ | 92.9 | 94.6 | 87.5 | 82.1 | 93.1 | 82.8 | 72.4 | 65.5 | 89.5 | 78.9 | 63.2 | n/a |

**Table 6.3.:** Overview of the classification results for the evaluations of two concatenated middle parts of test letters. Additional evaluations can be found in Tables B.5 to B.7 in Appendix B.1.3. General descriptions of the table structure and colours can be found in Table 6.1.

$N^{th}=1, H=5$, suffers from a too short history.

In the case of a segmentation of a series of observations into the underlying letters, a good compromise between the faithful classification of single points (using a long history of observations) and a limitation of the momentum-like effect (using a small history) therefore needs to be found.

It is important to notice that, whereas the evaluation is carried out for the example of the letter data set, the analysed issues constitute general aspects of the method. Since the compatibilities along with the history consideration are defined in terms of the manifolds, the transitions between classes or candidate motions and the momentum-like effect occur for all kinds of data that is represented in Structured UKR manifolds in the same way.

### 6.2.4. $3^{rd}$ Experiment: Segmentation of concatenated parts of letter trajectories

In the previous experiment, it has already been shown that a segmentation of a series of observations into the underlying letters can be performed on the basis of single point classifications. Whereas the previous experiment focussed on the history effects at letter transitions, the next experiment has rather the purpose to demonstrate the flexibility of the presented approach.

The third experiment therefore applies the segmentation approach to concatenated *parts* of the test sequences: half of the points in the middle of the trajectories (not according to time, but to the number of observations) are extracted from the original test sequences and are concatenated to new test combinations. Figure 6.8 depicts an example of such trajectory, where again the order in the sequence name corresponds to the order of the concatenated letter parts in the combination.

In comparison to the previous experiment, only half of the observations of each whole letter are used. Consequentially, only half of the motion process is observed. In addition, all sequence combinations do not start or end at the beginning or end of a represented candidate letter, and the transitions between the concatenated letters are also not at start

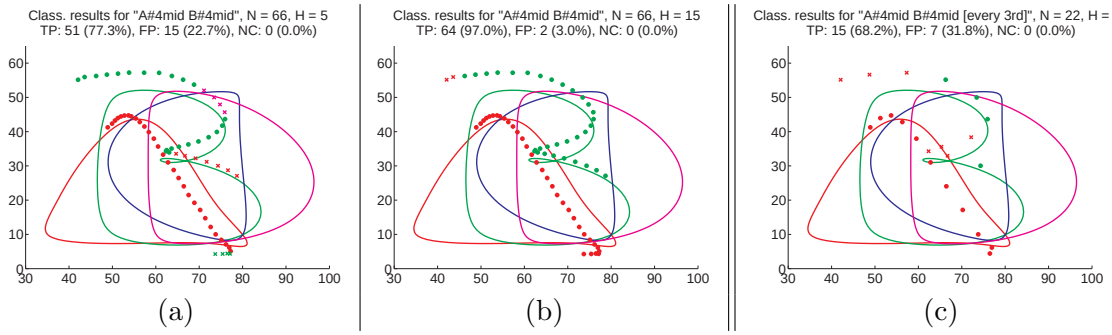



**Figure 6.8.:** Classification details for the sequence combination '$A_4^m B_4^m$' of two middle parts of the corresponding test letters. (a-b) depict evaluations of the fully sampled sequences ($N^{th} = 1$) for the history lengths (a) $H = 5$ and (b) $H = 15$. (c) shows the sparsely sampled trajectory ($N^{th} = 3$) also for $H = 15$. General descriptions of the plot colours and data can be found in Figure 6.5.

and/or end points of represented candidates. Both conditions render it more complicated for the classification system to extract the correct letter segmentations.

Again, the approach is evaluated for different sampling rates which are emulated by considering all test points ($N^{th} = 1$), only every second ($N^{th} = 2$) or every third ($N^{th} = 3$). The case $N^{th} = 4$ can be found in Appendix B.1.3.

Table 6.3 lists the corresponding classification results. The letter combinations are the same as in Table 6.2. The superscript '$^m$' in a sequence name indicates, that only the middle part of the original test sequence has been used.

The overall picture of the results appears to be below the results for the concatenated whole letters (see Table 6.2). Especially the sparser sampled trajectories and longer histories ($N^{th} = 2$ for $h = \{15, 20\}$ and $N^{th} = 3$ for $h = \{10, 15, 20\}$) yield rather poor rates of correct classifications. For the case $N^{th} = 1$, however, only those sequence combinations that involve trajectories with poor individual classification performance (see Table 6.1) yield significantly poorer results for the middle part sequences, i.e. '$C_3$' and '$C_5$', but also '$C_2$'. Otherwise, the results are very similar to the case of concatenated whole letter trajectories and even better for a number of combinations (e.g. '$A_1 B_1$', '$A_2 D_2$').

A closer inspection of single test results as depicted in Figure 6.8 provides insights into the reasons for the classification behaviour observed in Table 6.3. Figure 6.8 provides the results for the fully sampled sequence '$A_4^m C_4^m$' ($N^{th} = 1$) and the history lengths (a) $H = 5$, (b) $H = 15$, and the same sequence for a sparser sampling rate ($N^{th} = 3$), again for $H = 15$, in (c).

The consideration of the short history in Figure 6.8(a) yields the same effects already observed for the whole single letters and the concatenated whole letters: the trajectories are classified according to more local similarities to the candidate letters and the classification therefore changes the assigned class several times, especially in the '$B_4$' part of the sequence. Expectedly, the momentum-like effect at the letter transition point is very small and does not cause any wrong classified point at the beginning of the '$B_4$' trajectory.

By increasing the history length to $H = 15$, as depicted in Figure 6.8(b), a more global trajectory structure is captured and the classification results improve. However, the history momentum effect now also influences the classification and causes two misclassifications at the letter transition. Since the number of observations that describe the single letters is reduced to the half when considering only partial trajectories as described above, the momentum effect can become a critical issue. In Figure 6.8(c), a sparser sampled sequence

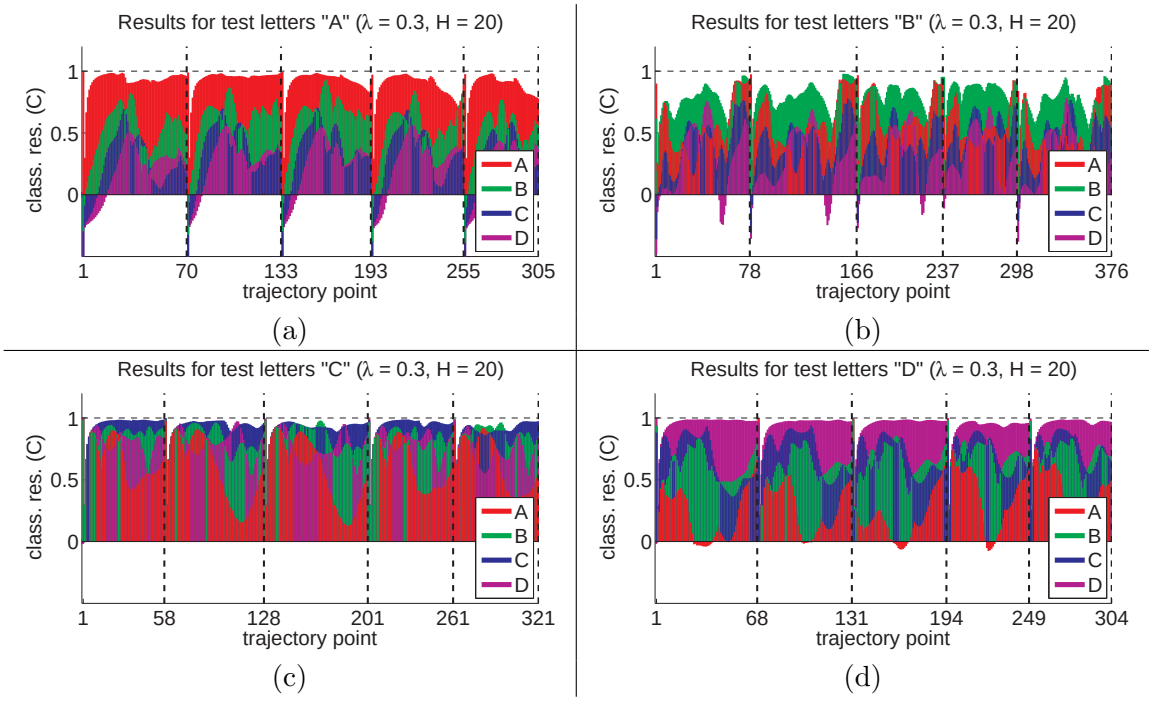



**Figure 6.9.:** Pointwise compatibility values $C$ (6.6) for the evaluations of the whole single test letters **(a)** '$A_1$', ..., '$A_5$', **(b)** '$B_1$', ..., '$B_5$', **(c)** '$C_1$', ..., '$C_5$', and **(d)** '$D_1$', ..., '$D_5$' using the parameters ($\lambda = 0.3$, $\gamma = 0.9$, and $H = 20$). The vertical dashed lines indicate the beginnings of new letters. The horizontal dashed line indicates the maximal compatibility value of $C = 1$. Plotted are the compatibilities with the UKR manifold representations of the letters '$A$' (red), '$B$' (green), '$C$' (blue), and '$D$' (magenta).

is evaluated with the same history length used in (b). In this case, the momentum effect causes 3 misclassifications, which however correspond to 13.64% of all observations and even to 27.27% of the $B_4$ part. Together with the four 'non-momentum' misclassifications, the overall classification result decreases to 68.18% true positives. For even shorter trajectory parts in the same context, this effect becomes more dominant since the total number of points reduces while the strength of the momentum effect stays the same.

### 6.2.5. On classification robustness and rejecting observed trajectories

One interesting issue in the context of motion recognition with candidate motions is the rejection of observed motions that do not fit to any of the candidates. In this context, Steffen, Pardowitz, and Ritter (2009b) suggested to use a threshold as lower bound for the compatibility value $C$. Below this bound, points could be rejected and classified as *unknown*. The trajectories evaluated by Steffen, Pardowitz, and Ritter (2009b) based on the same recordings that are used for the evaluation in this section. Since that work used the whole six dimensions from the recorded observations and the basis for the evaluation presented in this section are only the two-dimensional projections onto the 'virtual drawing plane' (see Section 6.1), the results differ significantly from each other.

Whereas the original idea of using only two dimensions was to provide more intuitive results that are easier to comprehend, it turned out that the four dimensions more in this dataset indeed hold information that facilitated to disambiguate the different test sequences.

The rather unexpected effect results from the way of recording the data where different letter classes have been recorded with characteristic hand orientations or hight levels. Since these effects are difficult to illustrate, the evaluation focussed on only the 2D trajectories that directly correspond to a 'virtual drawing plane'.

Figure 6.9 depicts the pointwise compatibility values $C$ (6.6) of all two-dimensional test sequences with full sampling (i.e. $N^{th} = 1$). For the computation of the depicted compatibilities, the parameters $\lambda = 0.3$, $\gamma = 0.9$, and $H = 20$ have been used.

Figure 6.9(a) visualises the results for the test sequences representing '$A$' trajectories (i.e. '$A_1$' to '$A_5$'), Figure 6.9(b) for those representing '$B$', Figure 6.9(c) for '$C$', and Figure 6.9(d) for '$D$', respectively. In the plots, the single letters are separated by vertical dashed lines.

Figure 6.9(a) illustrates that the compatibility measure achieves to distinguish well between the correct '$A$' classification and the other letters. The compatibility value $C$ is robustly higher for the '$A$' manifold than for all other candidates.

Figure 6.9(b) reveals, that the good classification rate for the '$B$' test sequences (see Table 6.1) are not based on very distinct differences of the compatibilities between the candidate representations. Still, the compatibility is highest for the '$B$' representation for most of the trajectory points and therefore results in correct classifications.

A similar, but less distinct situation can be observed in Figure 6.9(c) for the '$C$' test sequences. Whereas the compatibility with the '$C$' candidate is highest in the majority of observations for the first four test sequences (i.e. '$C_1$', '$C_2$', '$C_3$', and '$C_4$', respectively), the difference to the second best matching candidate is often only very small. The fifth test sequence '$C_5$' (see Section 6.2.2 and Figure 6.6) cannot be classified correctly for most of the observations.

The results for the '$D$' sequences depicted in Figure 6.9(d) are again robust, similar to the results for the '$A$' evaluations.

Apart from the insights about the robustness of the classifications and the distribution of the compatibilities for the evaluated test letters, Figure 6.9 also provides information about the ability to reject observations of unknown motions. In particular, the figures reveal that the usage of a lower bound for the compatibility value $C$ as a threshold for the rejection cannot yield perfect results for the underlying data.

For example, assuming the letter '$A$' not to be represented as candidate motion, the compatibility values in Figure 6.9(a) only exist for the candidates '$B$', '$C$', and '$D$'. These values constitute the basis on which the system needs to decide whether the trajectory parts are to be classified as one of these three candidates or rejected as *unknown*. In order to robustly reject the test sequences for the letter '$A$', however, a threshold for $C$ of 0.8 or higher needs to be applied. Such threshold would also cause the rejection of the major parts of the '$B$' test sequences depicted in Figure 6.9(b) which are classified correctly otherwise.

Applying the same considerations to the '$C$' candidate instead of '$A$', a threshold that robustly rejects letter '$C$' sequences is only slightly below the maximal compatibility value of $C = 1$ (see Figure 6.9(c)). This however would result in the rejection of almost all observations which are classified correctly using the winner-take-all mechanism.

The strategy of using a fixed threshold for the robust rejection of unknown sequences is therefore not viable with this system. At least for motions of a similar kind, like the four evaluated letters, the underlying compatibility measure $C$ (see Equation (6.6)) does not yield a sufficient disambiguation.

Figure 6.10 shows the compatibility results for the test sequences from Figure 6.9, but in the inverse chronological order of the observations, i.e. starting with the last observations

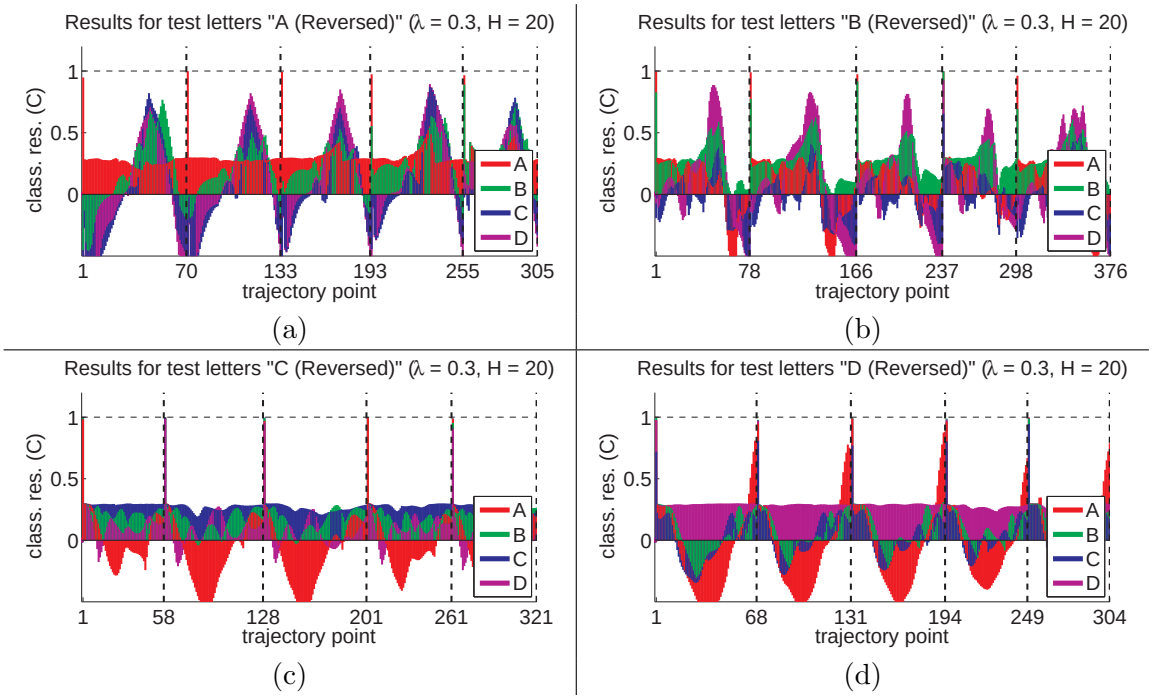



**Figure 6.10.:** Pointwise compatibility values $C$ (6.6) for the evaluations of the *reversed* whole single test letters **(a)** '$A_1^R$', ..., '$A_5^R$', **(b)** '$B_1^R$', ..., '$B_5^R$', **(c)** '$C_1^R$', ..., '$C_5^R$', and **(d)** '$D_1^R$', ..., '$D_5^R$' using the parameters ($\lambda = 0.3$, $\gamma = 0.9$, and $H = 20$). The vertical dashed lines indicate the beginnings of new letters. The horizontal dashed line indicates the maximal compatibility value of $C = 1$. Plotted are the compatibilities with the UKR manifold representations of the letters '$A$' (red), '$B$' (green), '$C$' (blue), and '$D$' (magenta).

and proceeding to the first original ones. Since the measure also considers the direction of the motion, all depicted sequences need to be rejected by the classification system. It is important to notice that the evaluated case corresponds on the one hand to the complement of the best matching case for the directional part of the measure, but on the other hand to the best matching case for the self-reconstruction part of it.

The compatibilities stay significantly below those of the sequences in correct chronological order. The maximal values yield the middle parts of the reversed '$A$' trajectories shown in Figure 6.10(a). For these trajectory parts, the projections of the reversed observations onto the candidates' manifold representations yield a correct chronological order for the beginning of the '$B$', '$C$', and '$D$' candidates. Although the observations yield relatively high self-reconstruction errors, the overall compatibility value reaches values of up to $C = 0.8$ for most of the '$A$' and some of the '$B$' test sequences, at least for a short part of the whole trajectory.

In all other cases, the compatibilities stay small for the major parts of the reversed test trajectories and can be rejected with a threshold of 0.5.

Figure 6.11 shows additional evaluations on artificial trajectory data in the centre range of the recorded test sequences. The figure is structured as follows: the first row depicts the test trajectories whereas the black points correspond to single observations which are connected by black lines in the underlying chronological trajectory order. The second and third row depict the corresponding pointwise compatibility values similar to Figure 6.9 and Figure 6.10, whereas the second row uses evaluation parameters ($\lambda = 0.3$, $\gamma = 0.9$, and

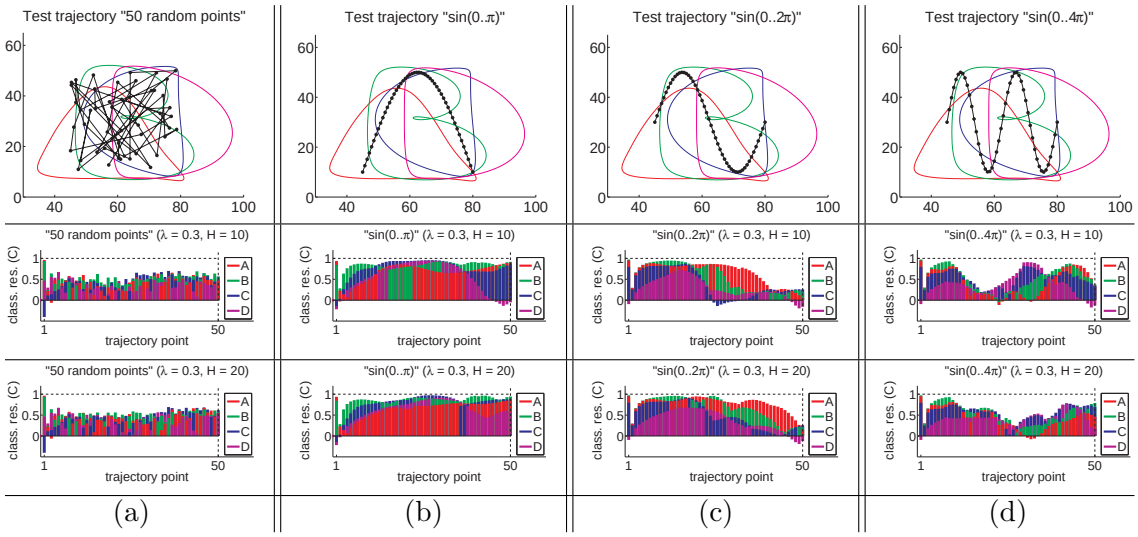



**Figure 6.11.:** Pointwise evaluations of sequences which are not represented in the candidate manifolds: **(a)** a trajectory of 50 random observations. **(b)** a trajectory representing a sine curve (0 to $\pi$). **(c)** a trajectory representing a sine curve (0 to $2\pi$). **(d)** a trajectory representing a sine curve (0 to $4\pi$). The first row depicts the evaluated trajectories together with the manifold representations of the candidate motions for the letters '$A$' (red), '$B$' (green), '$C$' (blue), and '$D$' (magenta). The second and third row visualises the pointwise compatibility values $C$ (6.6) for the trajectories of the first row. The second row uses the parameters ($\lambda = 0.3$, $\gamma = 0.9$, and $H = 10$) and the third row ($\lambda = 0.3$, $\gamma = 0.9$, and $H = 20$).

$H = 10$) and the third row ($\lambda = 0.3$, $\gamma = 0.9$, and $H = 20$), respectively.

In Figure 6.11(a), 50 random observations are considered. The maximal compatibility values are approximately at $0.5 \pm 0.1$. Whereas these values are unexpectedly high for a random trajectory, they still stay significantly below those for the real test sequences shown in Figure 6.9.

Figure 6.11(b) reveals a different picture for the results from the evaluation of a sine curve trajectory (50 points from 0 to $\pi$, scaled to the centre range of the real test data. The trajectory beginning is on the left side). Since the trajectory has similar characteristics as the real test letters, the pointwise compatibilities yield very high values throughout the trajectory. A rejection on the basis of a lower bound for the compatibility therefore is not possible.

Figure 6.11(c) holds the compatibility results for another sine curve, in this case from 0 to $2\pi$ (50 points, the trajectory starts on the left side). The compatibility values are in general smaller than in Figure 6.11(b), but the maximal values are still very high. Only for the shorter history of $H = 10$ (i.e. second row), the rear part of the trajectory yields a small compatibility with all candidates ($C < 0.4$) and can therefore be robustly rejected.

Figure 6.11(d) depicts the results for the trajectory of a sine curve from 0 to $4\pi$. The compatibility values are again smaller than those resulting from Figure 6.11(b-c), but still yield too high values for a robust rejection of the trajectory.

### 6.2.6. Discussion

In this chapter, an approach for the classification and segmentation of motion data has been presented that exploits the manifold features of the Structured UKR manifolds representing

the candidate motions. The basic approach has been presented in (Steffen, Pardowitz, & Ritter, 2009a) and (Steffen, Pardowitz, & Ritter, 2009b). The evaluation of the method has been largely extended and performed on a simplified, more illustrative data set. In addition to the evaluations provided by Steffen, Pardowitz, and Ritter (2009b), not only the classification results, but also the relation between winning class assignment and the compatibilities with the other candidate motions have been analysed.

Three experiments have been evaluated concerning the classification and segmentation abilities of the method. Section 6.2.2 addresses the general classification ability on sequences that consist of single whole letter trajectories. The evaluation yields promising results and shows that the method is able to classify the majority of test observations correctly.

In Section 6.2.3, the segmentation ability of the presented system is analysed. The evaluation of the classification performance on sequences consisting of two concatenated whole letter trajectories shows, that the method is in general only affected at the transitions between the two trajectory parts corresponding to different letters. At these points, the consideration of the history results in a momentum-like effect: at the beginning of the new letter, the considered history partly corresponds to the previous letter and therefore provides misleading history information. In consequence, the classification of such points yields the wrong class assignment. Since the momentum effect directly depends on the length of the history, it can be decreased by considering only short histories. However, since a longer history has been shown to be generally beneficial for the classification (see Section 6.2.2), a good compromise between the faithful classification of single points (using a long history of observations) and a limitation of the momentum-like effect (using a small history) needs to be found.

Section 6.2.4 evaluates the system's performance in segmenting concatenations of partial observations. Whereas the classification and segmentation results are still good in the majority of evaluations, the ratio between available observations per motion trajectory and the history length $H$ becomes more crucial. In particular, if the history length and the number of observations per letter are similar, the influence of the momentum effect becomes very strong and heavily decreases the classification rate (see Table 6.3, columns $N^{th} = 3, H \in \{15, 20\}$). In this context, a higher sampling rate that yields more samples per observed motion is important in order to reduce the relative influence of the momentum effect and therefore to enable the usage of a longer history.

Section 6.2.5 concerns the robustness of the pointwise classifications and the ability to reject observations as *unknown* (i.e. to classify the observations as not compatible with neither of the candidate motions) by using a fixed lower bound for the compatibility measure $C$. The results show that a winner-take-all mechanism indeed provides correct classifications in the majority of evaluations, but that the difference between maximal and second highest value sometimes is only very small. Evaluations of the chronologically reversed test sequences yield acceptable results (see Figure 6.10), but additional experiments on artificially created sine-curve-trajectories reveal that 'meaningful, but unknown' motions cannot be robustly recognised as such.

In order to keep the application robust, it should be therefore limited to the task of classifying and segmenting according to a known set of candidate motions. With this restriction, the method yield promising results.

To sum up, the two drawbacks of the presented method are the following:

- Due to the rather weak abilities in recognising unknown, previously not represented motions, a robust application of this system is limited to the classification and seg-

mentation of trajectories according to a set of represented candidate motions.

- In the case of sparsely sampled trajectories, the momentum effect described in Section 6.2.3 and 6.2.4 can become crucial. In consequence, for segmenting very short trajectory parts of different candidate motions, a high sampling rate is necessary.

On the other hand, the advantages of this approach are:

- Due to the definition of the compatibility features in the manifold domain instead of directly in the observation space, it is independent – to a certain extent – of the specific task or observation space characteristics.

- It can handle various sampling rates of the observations.

- It does not require a fixed number of observations and can therefore be applied on the 'raw' data stream without specific preprocessing in order to interpolate to specific sampling rates or observation counts.

- It does not require observations of complete motions, and it is also able to classify and segment trajectory parts.

- Since the computation of the compatibility measure only requires a (limited) history of observations, it is basically applicable to real time classification and segmentation of sensor data streams.

- Since the compatibilities to all candidate motions are evaluated, it can also be used to determine similar parts of different candidate motions or to provide a ranking of the most compatible candidates.

Another important benefit of the system is that, in principle, the ability to recognise (or classify or segment) a specific motion is automatically included in its representation as a Structured UKR manifold for reproduction and synthesis, because the recognition mechanism directly works on the basis of these reproduction manifolds.

Finally, it shall be mentioned again that, whereas the evaluation is carried out for the example of the letter data set, the analysed issues constitute general aspects of the method. Since the compatibilities along with the history consideration are defined in terms of the manifolds, the transitions between classes or candidate motions and the momentum-like effect occur for all kinds of data that is represented in Structured UKR manifolds in the same way.

In order to address the remaining drawbacks of the approach, an extension of the method is described in the next section. In particular, a competition of the candidate motions is introduced by incorporating the Competitive Layer Model (CLM; details, see next section). In addition, by introducing a modified compatibility measure and considering a symmetric chronological neighbourhood of the observations, the general classification abilities are improved. The influence of the momentum effect is largely decreased.

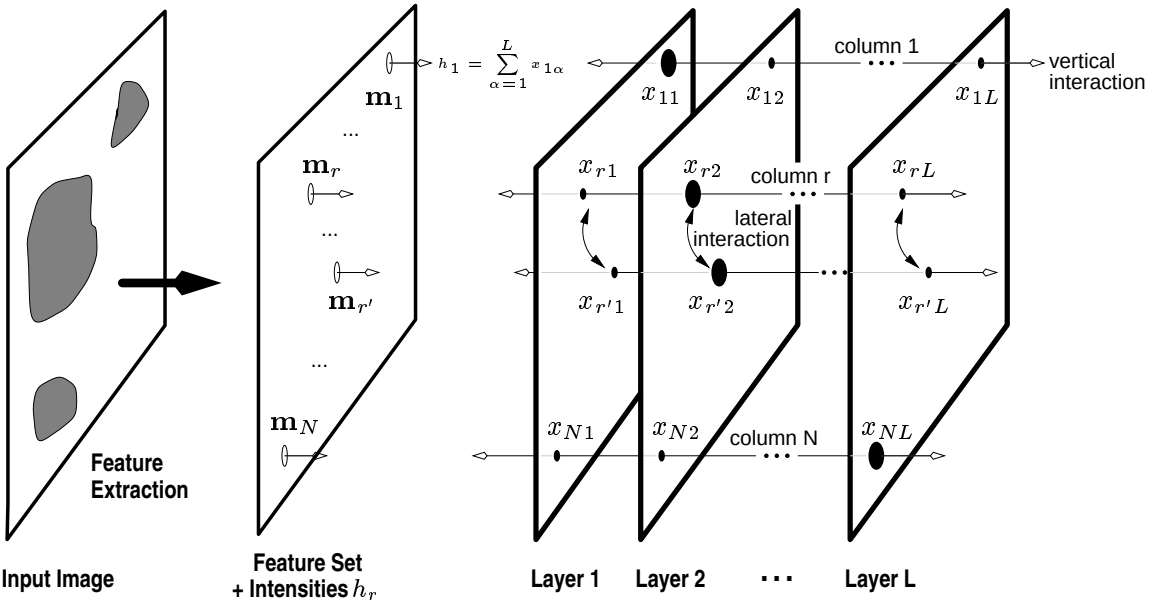



**Figure 6.12.:** CLM for perceptual grouping in images. Figure originally from (Ontrup & Ritter, 1998).

## 6.3. A combination of structured manifolds with a neural competitive layer model

### 6.3.1. The Competitive Layer Model (CLM)

The CLM is a recurrent neural network for grouping and segmentation of sensory input features. It has been introduced by Ritter (1990) for spatial feature linking, further applied to image segmentation and perceptual grouping (e.g. (Wersing, 2000; Wersing et al., 2001; Wersing, 2001; Nattkemper et al., 2002; Weng & Steil, 2002, 2003; Ontrup et al., 2004; Weng et al., 2006)) and recently transfered to action segmentation for robot task learning by Pardowitz et al. (2008, 2009).

The CLM realises a competition of feature assignment layers (each representing a feature group) of linear threshold neurons which encode contextual relations of input features as pairwise excitatory/inhibitory interactions (see Figure 6.12). These *compatibilities* define an energy function which rates the quality of the feature grouping and can be minimised with the neural dynamics of the CLM.

**The CLM architecture** Figure 6.12 depicts the general structure of the CLM: it consists of layers[3] $\alpha = 1..L$ each containing a fixed number of neurons. Each neuron within layer $\alpha$ has a fixed position $r$ and an associated non-negative activity $x_{r\alpha}$. *Columns* of activities $x_{r\cdot}$ of neurons at the same positions $r$ in different layers are associated with particular input vectors $\mathbf{m}_r$ of task-related feature values.

Afferent inputs $h_r$ can be included to express different levels of *significances* or *importance* of the feature $r$. Appropriate values for $h_r$ are related to the underlying task and need to

[3] In order to comply with the notation in the original works on the CLM, the symbols $\alpha$ and $\beta$ are overloaded and describe different CLM layers in this context.

be determined in a preprocessing step. They are fed to the activities $x_{r\alpha}$ with a connection weight $J_r > 0$.

Within one layer (or group), the pairwise compatibility (similarity/disparity) of two features $r$ and $r'$ is expressed by a lateral compatibility $f_{rr'}$. A pair that represents compatible features evokes mutual excitation ($f_{rr'} > 0$). On the other side, mutually incompatible features inhibit each other ($f_{rr'} < 0$).

Between the layers, columnar interactions $I_r^{\alpha\beta} = I_r^{\beta\alpha} > 0$ effect a mutual inhibition of neurons within the same feature column $r$ and realise a dynamical columnar winner-take-all circuit. This mechanism enforces exclusive assignments of features to one specific layer each.

Combining all ingredients into an additive activity dynamics yields

$$\dot{x}_{r\alpha} = -x_{r\alpha} + \sigma\left(J_r h_r - \sum_{\beta} I_r^{\alpha\beta} x_{r\beta} + \sum_{r'} f_{rr'}^{\alpha} x_{r'\alpha} + x_{r\alpha}\right) \tag{6.7}$$

$$= -x_{r\alpha} + \sigma\left(E_{r\alpha} + x_{r\alpha}\right). \tag{6.8}$$

The function $\sigma(x) = \max(0, x)$ ensures that the neuron activities $x_{r\alpha}$ are non-negative which is a requirement for the convergence of the dynamics (Wersing et al., 2001).

The corresponding *grouping* energy function that is used to rate the quality of a grouping/assignment can be denoted as

$$E = -\sum_{r\alpha} J_r h_r x_{r\alpha} + \frac{1}{2}\sum_{r}\sum_{\alpha\beta} I_r^{\alpha\beta} x_{r\alpha} x_{r\beta} - \frac{1}{2}\sum_{\alpha}\sum_{rr'} f_{rr'}^{\alpha} x_{r\alpha} x_{r'\alpha}. \tag{6.9}$$

Local minima of this energy are the attractors of the CLM dynamics (6.7) under the constraints $x_{r\alpha} \geq 0$. Since the negative value of the energy can be interpreted as the overall quality of the grouping, and the CLM dynamics converges to local minima of this energy, the dynamics results in (locally) optimal group assignments (Wersing et al., 2001).

**Incorporating annealing** In order to increase the grouping quality, a self-inhibition of each neuron can be incorporated using $f_{rr'}'^{\alpha} = f_{rr'}^{\alpha} - T\delta_{rr'}$ with $T > 0$. The modified CLM grouping energy can be denoted as:

$$E' = E + T\sum_{r\alpha} x_{r\alpha}^2. \tag{6.10}$$

The new term biases the local minima of the energy towards graded assignments with neuron activities $x_{r\alpha} > 0$ for several layers $\alpha$. By initialising the dynamics with a strong self-inhibition $T$ and decreasing it to 0 over time, an annealing process can be included in the grouping process. With vanishing 'temperature' $T$, the assignment behaviour changes from graded to unique assignments which constitute the final grouping result.

**Iterative solution of the CLM dynamics** A fast iterative solution of simulating the CLM dynamics (6.7) has been proposed by Ontrup and Ritter (1998) and Wersing (2000). It bases on the Gauss-Seidel method and realises a rapid search for fixed point attractors of the dynamics by iteratively solving the fixed point equations of (6.7) for a single, randomly chosen neuron activity $x_{r\alpha}$ (considering all other activities as constant). The pseudo-code of the algorithm has the following form:

1. Initialisation. Set all activities $x_{r\alpha}$ to small positive random values. Initialise $T$ with the highest eigenvalue of the matrices $\{f^{\alpha}_{rr'}\}$.

2. Solving. Repeat $N{\cdot}L$ times: choose a random tuple $(r, \alpha)$ in the borders of the underlying CLM and update:

$$\begin{aligned} x_{r\alpha} &= \max(0, \xi) \\ \xi &= \frac{1}{J_r - f_{rr} + T} \left( J_r h_r - \sum_{\beta \neq \alpha} I_r^{\alpha\beta} x_{r\beta} + \sum_{r' \neq r} f^{\alpha}_{rr'} x_{r'\alpha} \right) \end{aligned} \tag{6.11}$$

3. Annealing. Set $T = \eta T$ with $\eta \in [0; 1]$.

4. Convergence. If not converged, goto step 2.

**Summary of the important characteristics for motion recognition** The fundamental basis of the CLM is an appropriate preprocessing that extracts input features from the processed data and a corresponding function that provides a pairwise measure for input compatibilities $f_{rr'}$.

The CLM segments a set of $N$ input features into groups of mutually compatible features. Each group is represented by one CLM layer. Each layer provides one neuron for every input feature. The activities of these neurons express the assignments of the associated features to the corresponding layers.

During the grouping process, each layer competes for the exclusive assignment of all features that are compatible with it (and the rejection of the rest). Originally, this fixed compatibility association of specific feature characteristics with specific layers is usually not specified beforehand, but dynamically evolves from previous (partial) assignments of features to the same layer.

The grouping dynamics is driven by two main ingredients:

(a) an *intra-layer* pairwise compatibility negotiation between the assigned features and

(b) an *inter-layer* winner-take-all mechanism between the neurons from all layers that correspond to the same feature.

The CLM describes these two ingredients as terms of a *grouping energy* function, that provides a measure for the quality of a specific grouping result. Applying the CLM dynamics minimises this energy and it can be shown (Wersing et al., 2001) that it converges (under certain conditions) to stable fixed-points representing local minima of the grouping energy and thus to locally optimal groupings of the input features. The resulting groups consist of the features that exclusively provide non-zero neural activities in the same layer.

To sum up, the crucial issues in using the CLM are the data preprocessing and the definition of the compatibility function. In order to use the CLM for the purpose of motion recognition in the way it has been described in the previous section, these two issues are realised in terms of the UKR manifolds. By assigning one UKR manifold to each CLM layer and letting the CLM dynamics work on the features defined in terms of UKR, the grouping of compatible features corresponds to a dynamical compatibility determination according to the candidate motions. How such mechanism can be realised is detailed in the remainder of this chapter.

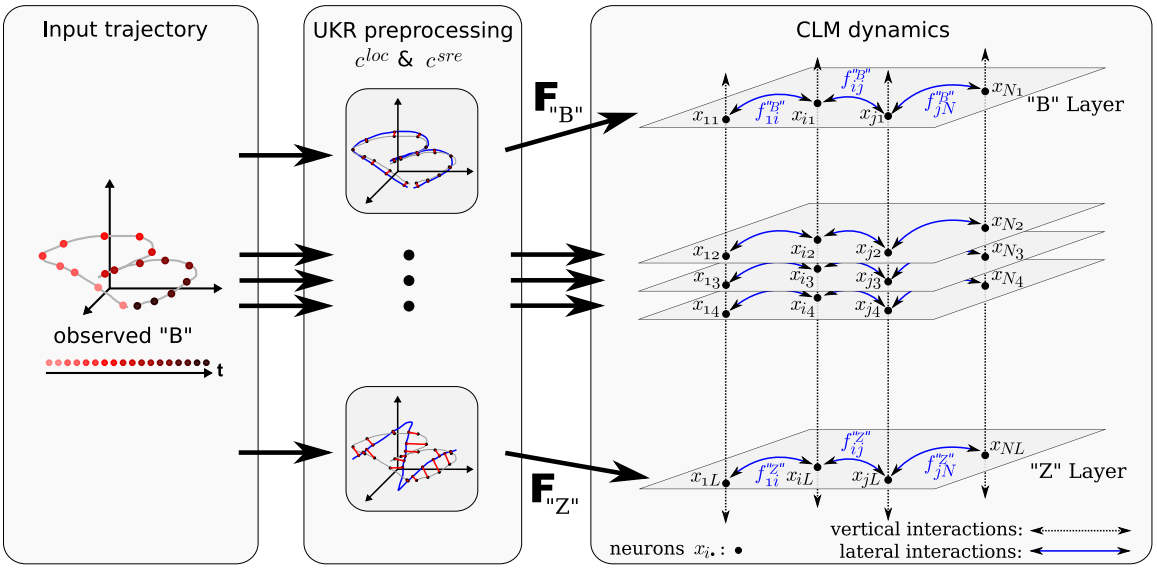



**Figure 6.13.:** Overview of the UKR/CLM scheme. The UKR manifolds are used to compute layer-specific compatibility matrices $\mathbf{F}$. The CLM dynamics is then used to find the compatible groups of features according to these compatibilities. The way of incorporating the input data is therefore different from the original approach (see Figure 6.12).

### 6.3.2. Combining UKR and CLM

In the combination of the two methods, the CLM performs the segmentation of observed trajectory samples according to a set of candidate patterns. These candidates are represented by UKR manifolds. The CLM input features of the observed trajectory, which are required by the CLM dynamics, are calculated on the basis of the UKR representations of these candidates. Figure 6.13 depicts an overview of the combined system.

Whereas the CLM has been designed in a general fashion, it usually uses global layer-independent features and therefore focusses on the pure pairwise mutual compatibilities of these features.

For the combination of the CLM with UKR manifolds and in order to segment according to a known set of candidate UKR models, every CLM layer is associated with one specific UKR manifold. The input features of the layers are computed on the basis of the layer-specific UKR manifolds. Every CLM layer therefore has its own input features and the CLM dynamics focusses on the segmentation of the features into coherent groups which are compatible with the assigned CLM layer.

The main issue of the UKR/CLM combination is to define both a preprocessing function and a compatibility measure *in terms of the UKR manifold*. This procedure decouples the CLM from the structure and the characteristics of the data itself, and only focusses on their UKR representations. Since these representations are designed to be unified for a broad range of applications, this constitutes a more general approach for any data represented in UKR manifolds.

The manifold features are inspired by the initial, purely manifold-based approach which did not include competition of the candidates (see Section 6.2). The features are based on:

(a) the order of the UKR latent representations $\hat{\mathbf{x}}_i = \mathbf{g}(\mathbf{y}_i)$ of the observed trajectory samples $\{\mathbf{y}_i\}$ and

(b) the normalised UKR self-reconstruction errors ('sre')

$$e_i^{\text{sre}} = (\mathbf{y}_i - \mathbf{f}(\hat{\mathbf{x}}_i))^T \boldsymbol{\Sigma}^{-1} (\mathbf{y}_i - \mathbf{f}(\hat{\mathbf{x}}_i)) \tag{6.12}$$

with $\boldsymbol{\Sigma}$ being the diagonal matrix of the dimension-wise UKR training data variances (see Section 2.1 for details on $\mathbf{f}(\cdot)$ and $\mathbf{g}(\cdot)$).

Due to the temporal context of the sequences, the mutual compatibilities are usually restricted to a limited time horizon by only considering a temporal neighbourhood $\mathcal{N}_i = \{\mathbf{y}_j \mid 0 < |i-j| < H\}$ of each input feature $\mathbf{y}_i$. $H$ is the neighbourhood parameter of the method: large values of $H$ focus on more global structures whereas small values are rather sensitive to local structures.

Feature (a) can be directly used as compatibility indicator of observed and represented motion: since the latent space of a Structured UKR manifold reflects the temporal order of the represented pattern, the values of the latent time dimension $d_t$ of latent projections $\hat{\mathbf{x}}_i$ of compatible observations are in correct chronological order. Vice versa, observations whose latent projections are not correctly ordered are likely to be incompatible.

Functions that realise a corresponding local order measure have already been presented in Section 6.2 (see Figure 6.3) in the context of the purely manifold-based recognition approach. In the case of the CLM, the periodic step function depicted in Figure 6.4(b) turned out to be a good choice:

$$H_{\circlearrowleft}^{\pi}(x) = \text{sgn}(\cos(\text{mod}_{\pi}(x))) \tag{6.13}$$

On the basis of $H_{\circlearrowleft}^{\pi}(\cdot)$, the *latent order compatibility (loc)* of two observations $\mathbf{y}_i$ and $\mathbf{y}_j$ $(i < j; j \in \mathcal{N}_i)$ can be denoted as:

$$c_{ij}^{\text{loc}} = c_{ji}^{\text{loc}} = H_{\circlearrowleft}^{\pi}((\hat{\mathbf{x}}_j)_{d_t} - (\hat{\mathbf{x}}_i)_{d_t}) \qquad \in [-1;1] \tag{6.14}$$

and $\forall i \colon c_{ii}^{\text{loc}} = 0$.

Feature (b) can also be used directly as compatibility measure (see Section 6.2). However, considering the average self-reconstruction error in the temporal neighbourhood $\mathcal{N}_j$ of the evaluated $\mathbf{y}_j$ turned out to be more robust (again for observations $\mathbf{y}_i$ and $\mathbf{y}_j$, whereas $i < j$ and $j \in \mathcal{N}_i$):

$$c_{ij}^{\text{sre}} = c_{ji}^{\text{sre}} = -1 + 2\exp\left(-\frac{1}{|\mathcal{N}_j|}\sum_{k\in\mathcal{N}_j}\left\|\frac{\Delta_e(k)}{\Delta_{\hat{\mathbf{x}}}(k)}\right\|\right) \qquad \in [-1;1] \tag{6.15}$$

where $\Delta_e(k) = e_k^{\text{sre}} - e_{k-1}^{\text{sre}}$ and $\Delta_{\hat{\mathbf{x}}}(k) = \hat{x}_{k,d_t} - \hat{x}_{k-1,d_t}$ with $d_t$ being the latent time dimension and $\Delta_e(1) = \Delta_{\hat{\mathbf{x}}}(1) = 0$.

The matrix $\mathbf{F}$ of CLM compatibilities can be denoted as a balanced combination of both measures.

In order to express the connections between the CLM compatibilities $f_{rr'}$ (see Section 6.3.1) and the involved observations $\mathbf{y}_i$ and $\mathbf{y}_j$ along with their UKR latent representations $\mathbf{x}_i$ and $\mathbf{x}_j$, the CLM compatibilities will be referred to as $f_{ij}$ in the context of UKR, denoting the compatibility of observations $\mathbf{y}_i$ and $\mathbf{y}_j$, respectively.

The components of the new CLM compatibility matrix $\mathbf{F}$ are therefore denoted as:

$$(\mathbf{F})_{ij} = f_{ij} = \frac{1}{2}(c_{ij}^{\text{loc}} + c_{ij}^{\text{sre}}) \qquad \in [-1;1] \tag{6.16}$$

describing the compatibility of the observations $\mathbf{y}_i$ and $\mathbf{y}_j$ with each other and with the underlying layer-corresponding UKR manifold.

An overview of the whole system is given in Figure 6.13. The compatibilities $c^{\text{loc}}$ and $c^{\text{sre}}$ and therefore the matrices $\mathbf{F}$ are calculated during the UKR preprocessing (see Figure 6.13, centre box).

### 6.3.3. The CLM 'background layer'

In the domain of perceptual grouping in image processing, the CLM has been endowed with an additional layer for the purpose of attracting noisy features that do not correspond to a salient group of input features (e.g. (Wersing, 2000)). Since such features mostly appear in the background of an image, the layer is also denoted as "background layer". It usually consists of compatibilities $f_{ii}^{bg} = \lambda_{bg}$ that do not depend on the underlying image data, but describe fixed self-excitations of the corresponding neurons with parameterisable strength. The resulting effect is that only those features $i$ are assigned to a salient group layer that provide a sufficiently high support by a sufficiently high number of other compatible features $j$. Features that cannot provide an overall support above the threshold defined by the background strength are automatically assigned to the background layer.

In the domain of motion recognition and segmentation, such mechanism is also very desirable. Especially in the context of a fixed set of candidate motions, the background layer can be useful in order to decide if the observed motion corresponds to one of these candidates, or if it constitutes a new, previously unknown motion.

Whereas the neighbourhood of neurons within a CLM layer was not necessarily meaningful in the original approach, it directly corresponds to the chronological proximity of the associated observations in the motion recognition context. This consideration can be used as a basis for a background layer that also influences the minimal length of a recognised motion part. Such effect can be realised by providing not only self-excitations $f_{ii}^{bg} = \lambda_{bg}$ of the background neuron associated with observation $\mathbf{y}_i$ (and in particular $f_{ij}^{bg} = 0$, if $i \neq j$), but also supporting neighbouring neurons (associated with observations $\mathbf{y}_j$) with an excitation $f_{ij}^{bg} > 0$ for a certain chronological proximity $|i - j| < K_{bg}$.

In order to incorporate a smoother background behaviour, a decreasing support with higher chronological distance is used:

$$f_{ij}^{bg} = \begin{cases} \frac{1}{1+|i-j|}\lambda_{bg} & \text{if } |i-j| < K_{bg} \\ \\ 0 & \text{else} \end{cases} \tag{6.17}$$

The background strength $\lambda_{bg}$ and the background neighbourhood radius $K_{bg}$ are parameters to the background layer and can be set according to the requirements of the task.

### 6.3.4. Evaluation

The evaluation focusses on the recognition and segmentation abilities of the combined CLM/UKR system. Whereas the method is generally applicable to all kinds of sequence data that is represented in Structured UKR manifolds, also the combined UKR/CLM approach is evaluated on the 'ABCD' dataset described in Section 6.1, in order to allow for comparing the results of both methods.

The training, validation and test data consist of the same data that has been used for the UKR-only evaluation described in Section 6.2.1. The data is visualised in Figure 6.1. Each

set consists of five trajectories for each of the letters 'A', 'B', 'C', and 'D'. Each trajectory corresponds to the path of a tracked hand drawing a letter in the air and is extracted from monocular camera pictures from an orthogonal view onto the virtual drawing plane.

In preparation of the evaluation, one Structured UKR manifold for each of the letters is trained, each on the basis of the five training sequences of the corresponding letter and each with a one-dimensional periodic latent space. In order to compare the results of the following evaluation with the results of the pure manifold approach to motion recognition, the same UKR manifolds have been used (see Section 6.2.1). The represented letter trajectories are depicted in Figure 6.1. Each trained manifold is associated with one of four CLM layers. An additional layer constitutes the background layer as described in the previous section. The optimisation of the model parameters has been performed on the validation data set. Afterwards, the evaluation presented in the following is based on the test data set.

The evaluation of a test sequence consisting of observed trajectory samples $\{\mathbf{y}_i\}, i = 1 \dots N$ is performed as follows:

1. Each CLM layer is equipped with an ordered set of $N$ neurons corresponding to the $N$ observation samples. Each observation $\mathbf{y}_i$ therefore has a five-dimensional neural representation in the CLM. The first four components constitute the neural activities of the candidate letters represented by the layer-specific UKR manifolds. The fifth component corresponds to the activity of the corresponding background neuron. All neurons are initialised with small positive random activations.

2. For each CLM layer, the layer-specific $N \times N$ compatibility matrix $\mathbf{F}$ (6.16) is computed on the basis of the layer-associated manifold (see Section 6.3.2 and 6.3.3). For the background layer, the parameters $\lambda_{bg} = (1-10/N)$ (for $N > 10$) and $K_{bg} = H$ are used (if nothing else is declared), where $H$ corresponds to the neighbourhood radius used in the specific evaluations. The strength $\lambda_{bg}$ of the background layer needs to be weaker for sparser sampled trajectories. Since the sampling rate is emulated by considering every, every second, and every third data point of the test data, such behaviour has been realised depending on the total number $N$ of trajectory observations during the validation phase and is kept fixed for the evaluation on the test set.

3. The CLM dynamics is applied (using the iterative approach described in Section 6.3.1) until convergence, i.e. until only one non-zero neural activity remains in the 5D neural representation of any $\mathbf{y}_i$, where each component corresponds to the activity of one CLM layer and therefore to one letter or the background layer. Since no assumptions about the significance of different observations can be made, the CLM input significance $h_r$ is set to 1 for all observations. Likewise, the significance weightings $J_r$ are set to 2 for all observations. These parameters yielded promising results for the validation data set. Afterwards, all observations are uniquely assigned to one CLM layer and to one UKR representation of a candidate letter, respectively. These assignments constitute the final pointwise classification results.

In order to compare the combined CLM/UKR system with the purely UKR-based approach described in Section 6.2, the same experiments are performed as have been presented in Section 6.2.1.

One additional experiment focusses on the effects of the background mechanism of the CLM described in the previous section. This mechanism constitutes an active pendant to the passive compatibility threshold that is featured by the purely manifold-based approach.

| | $N^{th} = 1$ | | | | $N^{th} = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1$ | 87.9 (8.3) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_2$ | 86.5 (15.6) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_3$ | 73.3 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_4$ | 78.6 (12.1) | 98.4 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_5$ | 68.0 (0.0) | 70.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 68.0 (0.0) | 100.0 (0.0) | 69.0 (10.0) | 64.0 (0.0) |
| $B_1$ | 87.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 79.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_2$ | 84.9 (0.6) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 96.6 (1.3) | 97.7 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_3$ | 91.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 94.4 (0.0) |
| $B_4$ | 94.3 (11.5) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_5$ | 64.1 (0.0) | 69.6 (3.2) | 69.2 (0.0) | 70.5 (0.0) | 69.9 (3.2) | 38.5 (0.0) | 51.3 (0.0) | 48.7 (0.0) |
| $C_1$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_5$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_1$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_5$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |

**Table 6.4.:** Overview of the classification results for the evaluations of whole single letter sequences using the combined UKR/CLM system. The sequence name (first column) indicates the evaluated letter ('A', 'B', 'C', and 'D') and the corresponding test trajectory number ($1-5$ for each letter). The sequences are evaluated for two different sampling rates, either considering *every* observation point ($N^{th} = 1$) or *every second* ($N^{th} = 2$). Evaluation results for $N^{th} = 3$ and $N^{th} = 4$ can be found in Table B.8 in Appendix B.2.1. For each sampling rate, evaluations are performed for four different history lengths, i.e. $H = 5, 10, 15$, and 20. The listed values constitute the percentage of true positive point classifications for the processed sequence. The first value corresponds to the mean and the second value (in parentheses) to the standard deviation of four trials: mean (stddev). The background colours of the table cells encode the underlying mean classification result, whereas green corresponds to good results and red to very poor ones. Due to the random initialisation of the neuron activities of the CLM layers and a non-deterministic part in the procedure to run the CLM dynamics, every evaluation is done four times.

### 6.3.5. $1^{st}$ Experiment: Classification of whole-letter-trajectories

The first experiment evaluates the method concerning the classification of trajectories of single and complete letter trajectories for 'A', 'B', 'C', or 'D', respectively. Since the evaluation of the CLM dynamics includes non-deterministic parts (i.e. the random initialisation of the neuron activities as well as the Gauss-Seidel method), each sequence is evaluated in four trials.

In direct correspondence to the evaluation in Section 6.2.2, Table 6.4 provides an overview of the results of all evaluations. All sequences are evaluated using different sampling rates $N^{th} = 1, 2$. The cases $N^{th} = 3$ and $N^{th} = 4$ are omitted here and can be found in Appendix B.2.1. Cell colours encode the displayed mean classification results using the same colour scale previously applied to the UKR-only evaluation. The listed values constitute the percentage of true positive point classifications for the processed sequence. The first value corresponds to the mean classification rate and the second value (in parentheses) to the standard deviation of four trials: mean (stddev).

The overall result is significantly improved compared to the manifold-only case: 143 of 160

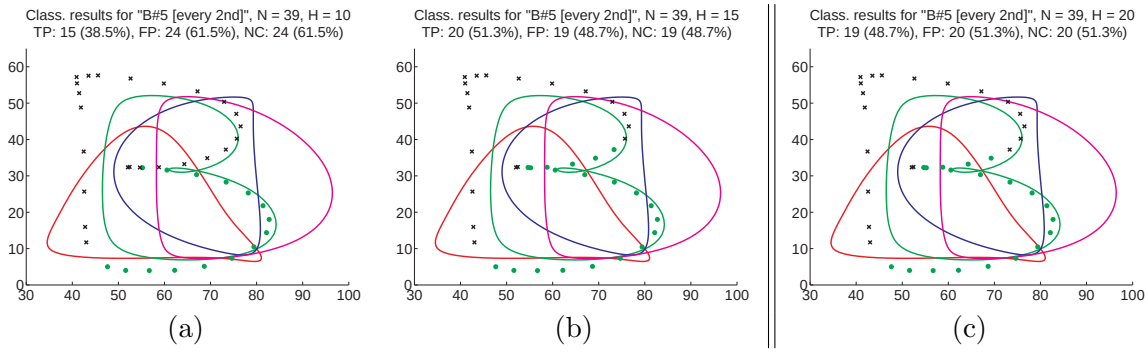



**Figure 6.14.:** Classification results for sequence '$B_5$' for $N^{th} = 2$ and $H = 10, 15, 20$. The depicted test cases yielded the worst results with the combined UKR/CLM system. Black crosses correspond to observations that have been assigned to the CLM background layer and have therefore been rejected. General descriptions of the plot colours and data can be found in Figure 6.5.

evaluated sequences (89.38%) yield average classification rates above 85% (in comparison to 108 of 160 (67.50%) without CLM for $N^{th} \in \{1, 2\}$, see Table 6.1). In 135 of 160 cases (84.38%), 100% of the observations could be classified correctly.

The system yields also very good classification results for the sequences '$C_3$' and '$C_5$' which provide poor results with the UKR-only system (see Table 6.1 and especially Figure 6.6).

In contrast to that, especially test sequence '$B_5$' and also partly '$A_5$' perform worse with the combined UKR/CLM system than before. Figure 6.14 shows the test cases for the sequence '$B_5$' that yielded the worst results listed in 6.4, i.e. $N^{th} = 2$ for $H = 10, 15$, and 20. Black crosses correspond to observations that have been assigned to the CLM background layer and have therefore been rejected. In all depicted cases, about the first half of the observations (the data points around the upper left corner of the letter) are rejected. Interestingly, as listed in Appendix B.2.1, the same sequence for sparser sampling rates, i.e. $N^{th} = 3$, and 4, yield correct classification rates of 95% and higher for most of the test cases. This indicates that the first half of the letter '$B_5$' is indeed most compatible with the '$B$' representation, but the background strength determined from the validation data requires more similarity in order to prevent a rejection, at least for the denser sampled trajectories.

Concerning the behaviour of the CLM background layer, the observed behaviour can be seen as confirmation of the proper functioning of the rejection mechanism. Whereas the background strength was optimised for the validation data, which is more similar to the '$B$' representation in the upper left corner of the letter (see Figure 6.1), the same background strength results in the rejection of the less similar version in the test data. For the sparser sampled trajectories, the background strength is reduced apparently to a level that allows for the assignment of the formerly rejected observations to the correct class. On the one hand, this indicates that the mechanism to choose the background strength $\lambda_{bg}$ depending on the number of observations $N$ (i.e. $\lambda_{bg} = (1 - 10/N)$), that had been chosen for evaluation purposes and due to the mechanism of emulating different sampling rates, does not perform perfectly. On the other hand, it also indicates that the background strength can be intuitively adapted to the task requirements.

| | $N^{th} = 1$ | | | | $N^{th} = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1B_1$ | 90.2 (3.4) | 89.0 (0.3) | 94.6 (0.0) | 100.0 (0.0) | 87.8 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 87.8 (0.0) |
| $A_1C_1$ | 93.0 (4.7) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 98.4 (0.0) | 92.2 (0.0) |
| $A_1D_1$ | 95.1 (4.9) | 98.6 (0.0) | 96.4 (0.0) | 94.2 (0.0) | 98.6 (0.0) | 94.2 (0.0) | 91.3 (0.0) | 87.0 (0.0) |
| $A_2B_2$ | 78.8 (0.0) | 98.0 (0.0) | 96.7 (0.0) | 94.0 (0.0) | 98.7 (0.0) | 93.8 (0.7) | 96.1 (0.0) | 98.7 (0.0) |
| $A_2C_2$ | 89.8 (0.8) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 97.8 (2.6) |
| $A_2D_2$ | 95.0 (5.7) | 98.4 (0.0) | 96.8 (0.0) | 94.4 (0.0) | 100.0 (0.0) | 95.3 (0.0) | 92.2 (0.0) | 87.5 (0.0) |
| $A_3B_3$ | 84.2 (6.5) | 97.7 (0.0) | 96.9 (0.0) | 100.0 (0.0) | 98.5 (0.0) | 95.5 (0.0) | 92.4 (0.0) | 86.4 (0.0) |
| $A_3C_3$ | 91.7 (5.7) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 86.6 (0.0) |
| $A_3D_3$ | 90.2 (6.5) | 97.6 (0.0) | 95.9 (0.0) | 93.5 (0.0) | 98.4 (0.0) | 95.2 (0.0) | 90.3 (0.0) | 85.5 (0.0) |
| $A_4B_4$ | 82.9 (0.0) | 97.8 (0.8) | 95.1 (0.0) | 97.6 (0.0) | 85.5 (0.0) | 98.4 (0.0) | 88.7 (0.0) | 80.6 (0.0) |
| $A_4C_4$ | 89.1 (6.1) | 98.4 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 96.7 (0.0) | 83.6 (0.0) |
| $A_4D_4$ | 91.9 (7.4) | 97.0 (0.5) | 95.7 (0.0) | 94.0 (0.0) | 98.3 (0.0) | 94.9 (0.0) | 91.5 (0.0) | 86.4 (0.0) |
| $A_5B_5$ | 71.9 (0.0) | 74.2 (0.0) | 81.3 (0.0) | 80.9 (0.5) | 73.4 (0.0) | 62.5 (0.0) | 53.1 (0.0) | 54.7 (0.0) |
| $A_5C_5$ | 85.5 (0.0) | 85.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 85.5 (0.0) | 100.0 (0.0) | 83.6 (0.0) | 81.8 (0.0) |
| $A_5D_5$ | 84.8 (0.0) | 96.2 (0.0) | 93.3 (0.0) | 84.8 (0.0) | 84.9 (0.0) | 92.5 (0.0) | 83.0 (0.0) | 83.0 (0.0) |

**Table 6.5.:** Overview of the classification results for the evaluations of two concatenated whole letter trajectories using the combined UKR/CLM system. Additional evaluations can be found in Tables B.9 to B.11 in Appendix B.2.2. General descriptions of the table structure and colours can be found in Table 6.4.

### 6.3.6. $2^{nd}$ Experiment: Segmentation of concatenated whole letter trajectories

The second experiment addresses the segmentation of series of observations according to the set of candidate classes. The evaluation context corresponds to the second experiment with the UKR-only system described in Section 6.2.3. The CLM features are again defined for every single observation under consideration of a certain chronological neighbourhood. In the UKR-only case, this condition decouples most of the single point classifications from effects introduced by a concatenation of several trajectories. Due to the CLM dynamics, such effects however can be propagated through the whole sequence during the course towards convergence of the dynamics. Apart from this new issue, the main focus of the evaluation still lies on the classification behaviour of the single point classifications at transition points between two letters. These transitions involve the observations for which the competition between the two letters can be expected to be strongest.

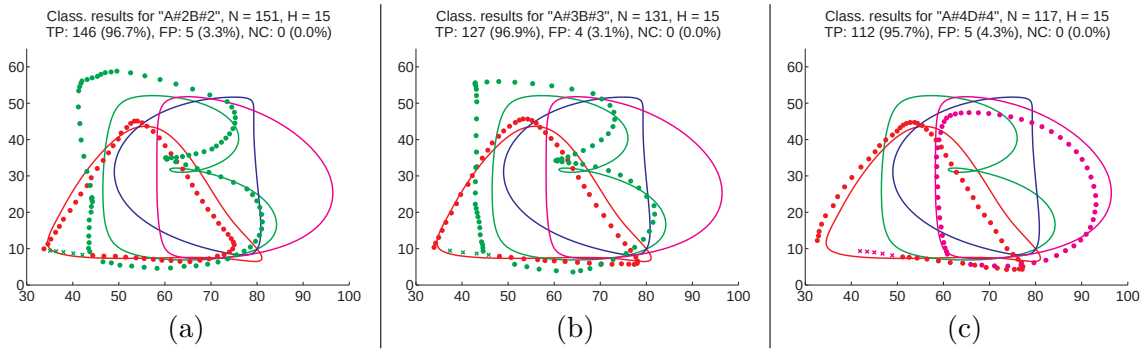



**Figure 6.15.:** Evaluation details for the combined sequences (a) '$A_2B_2$', (b) '$A_3B_3$', and (c) '$A_4D_4$', each for the history length $H = 15$ and full sampling rate ($N^{th} = 1$), using the combined UKR/CLM system. Related results from the UKR-only method can be found in Figure 6.7. General descriptions of the plot colours and data can be found in Figure 6.5.

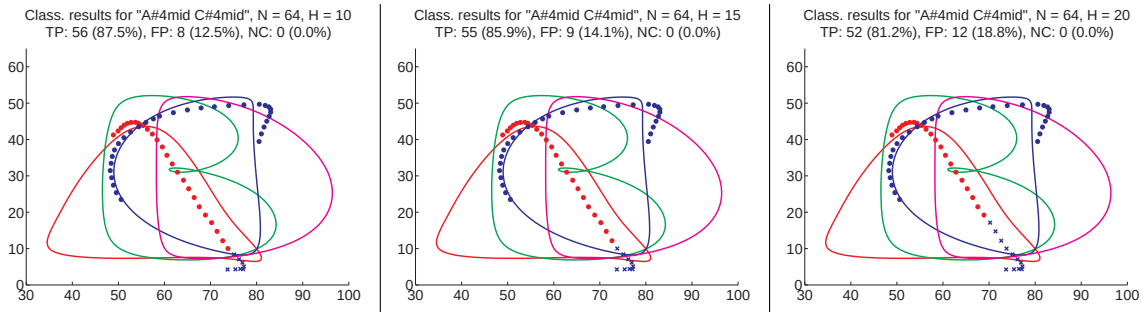



**Figure 6.16.:** Classification details for the sequence combination '$A_4^m C_4^m$' of two middle parts of the corresponding test letters, using the combined UKR/CLM system. The depicted evaluations are made on fully sampled sequences ($N^{th} = 1$) for the history lengths (a) $H = 10$, (b) $H = 15$, and (c) $H = 20$. General descriptions of the plot colours and data can be found in Figure 6.5.

Table 6.5 lists the evaluation results for combinations of the letter '$A$' test sequences with the '$B$', '$C$', and '$D$' test sequences. The names of the sequences listed in the table indicate the order of the concatenated whole letter sequences, e.g. '$A_1B_1$' is a sequence consisting of the whole test sequence '$A_1$' followed by the whole test sequence '$B_1$'. The evaluated combinations correspond to those used in Section 6.2.3 for the same evaluation with the UKR-only system. Additional evaluation can be found in Appendix B.2.2.

The general point classification performance of the system decreased in comparison to the single letter trajectories (see Table 6.4) which is consistent with the results for the UKR-only approach (see Table 6.1 versus Table 6.2). The overall performance of the UKR/CLM system is however still robustly good for the majority of evaluated letter combinations. 100 of 120 evaluations (83.33%) yield an average classification rate above 85%.

Whereas the transitions between letters are again the most prominent positions for false classifications (see Figure 6.15, the ends of the '$A$' trajectories), the effect is less strong than the momentum effect in the UKR-only case. In addition, for 29 of 120 letter combinations (24.17%), a perfect classification of all observations of both letter trajectories can be achieved. Such 100% true positive classifications are not present in the corresponding UKR-only evaluation. The involvement of the neural competition between the candidate classes, which is performed by the CLM dynamics, yields therefore advantages for the segmentation of trajectories.

### 6.3.7. $3^{rd}$ Experiment: Segmentation of concatenated parts of letter trajectories

The third experiment applies the segmentation approach to concatenated *parts* of the test sequences: half of the points in the middle of the trajectories (not according to time, but to the number of observations) are extracted from the original test sequences and are concatenated to new test combinations. The evaluation corresponds to the third experiment with the UKR-only system presented in Section 6.2.4.

Figure 6.16 depicts an example of such trajectory, whereas again the order in the sequence name corresponds to the order of the concatenated letter parts in the combination.

Table 6.6 lists the corresponding classification results. The letter combinations are the same as in Table 6.5. The superscript '$^m$' in a sequence name indicates, that only the middle part of the original test sequence has been used. Additional evaluation can be found in Appendix B.2.3.

| | $N^{th} = 1$ | | | | $N^{th} = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1^m B_1^m$ | 87.2 (0.0) | 95.8 (0.6) | 93.3 (0.6) | 89.7 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 96.3 (4.3) | 92.5 (0.0) |
| $A_1^m C_1^m$ | 88.2 (0.0) | 86.8 (0.0) | 83.8 (0.0) | 80.9 (0.0) | 91.4 (0.0) | 85.7 (0.0) | 77.1 (0.0) | 71.4 (0.0) |
| $A_1^m D_1^m$ | 87.5 (0.0) | 86.1 (0.0) | 83.3 (0.0) | 81.9 (0.0) | 86.5 (0.0) | 81.1 (0.0) | 73.0 (0.0) | 64.9 (0.0) |
| $A_2^m B_2^m$ | 87.8 (0.7) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 87.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_2^m C_2^m$ | 91.4 (0.0) | 89.6 (0.7) | 87.1 (0.0) | 84.3 (0.0) | 91.7 (0.0) | 84.0 (1.4) | 72.2 (0.0) | 75.0 (0.0) |
| $A_2^m D_2^m$ | 89.4 (0.0) | 87.9 (0.0) | 84.8 (0.0) | 81.8 (0.0) | 87.5 (1.5) | 82.4 (0.0) | 72.8 (1.5) | 64.7 (0.0) |
| $A_3^m B_3^m$ | 83.8 (0.0) | 92.6 (0.0) | 94.1 (0.0) | 91.2 (0.0) | 97.1 (0.0) | 91.4 (0.0) | 88.6 (0.0) | 65.0 (1.4) |
| $A_3^m C_3^m$ | 89.3 (7.1) | 91.4 (0.0) | 90.0 (0.0) | 78.6 (0.0) | 91.7 (0.0) | 77.8 (0.0) | 77.8 (0.0) | 80.6 (0.0) |
| $A_3^m D_3^m$ | 90.6 (0.0) | 90.6 (0.0) | 87.5 (0.0) | 82.8 (0.0) | 90.9 (0.0) | 84.1 (1.5) | 69.7 (0.0) | 63.6 (0.0) |
| $A_4^m B_4^m$ | 89.4 (0.0) | 92.4 (0.0) | 92.4 (0.0) | 94.3 (0.8) | 97.1 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_4^m C_4^m$ | 90.6 (0.0) | 87.5 (0.0) | 85.9 (0.0) | 81.3 (0.0) | 87.9 (0.0) | 82.6 (1.5) | 72.7 (0.0) | 72.7 (0.0) |
| $A_4^m D_4^m$ | 90.3 (0.0) | 88.7 (0.0) | 85.5 (0.0) | 80.6 (0.0) | 89.8 (1.6) | 87.5 (0.0) | 71.9 (0.0) | 65.6 (0.0) |
| $A_5^m B_5^m$ | 80.9 (0.0) | 77.9 (0.0) | 64.3 (0.7) | 54.4 (0.0) | 82.9 (0.0) | 60.0 (0.0) | 31.4 (0.0) | 31.4 (0.0) |
| $A_5^m C_5^m$ | 93.1 (0.0) | 93.1 (0.0) | 89.7 (0.0) | 82.8 (0.0) | 93.3 (0.0) | 83.3 (0.0) | 76.7 (0.0) | 76.7 (0.0) |
| $A_5^m D_5^m$ | 92.9 (0.0) | 89.3 (0.0) | 85.7 (0.0) | 78.6 (0.0) | 89.7 (0.0) | 79.3 (0.0) | 69.0 (0.0) | 51.7 (0.0) |

**Table 6.6.:** Overview of the classification results for the evaluations of two concatenated middle parts of test letters using the combined UKR/CLM system. Additional evaluations can be found in Tables B.12 to B.14 in Appendix B.2.3. General descriptions of the table structure and colours can be found in Table 6.4.

The results are generally below those for the concatenated whole letters, but constitute still very good pointwise classification rates in most of the cases. Most of the classification results are significantly better than in the corresponding UKR-only evaluation listed in Table 6.3.

A closer inspection of the detailed pointwise classification results provide further insights about the behaviour of the combined CLM/UKR system in comparison to the UKR-only case. Figure 6.16 exemplarily depicts such detailed results for the sequence '$A_4^m C_4^m$' for the neighbourhood radii $H = 10, 15$ and 20 and the sample rate $N^{th} = 1$. Whereas the letter transition point also yields classification errors, the general classification behaviour of the combined CLM/UKR system is more robust than the UKR-only system. In particular, for the test sequence '$A_4^m C_4^m$', the misclassifications at the letter transition constitute the only errors and all other observations are classified correctly. Due to this fact, the length of the correctly classified series of observations is still very large. For the robust segmentation, this is especially beneficial and enables the method to correctly recognise the main parts of the letters.

### 6.3.8. On classification robustness

The CLM dynamics converges to unique assignments of features (and therefore observations) to one layer (and therefore candidate motion). Due to these unique assignments, the results correspond to final, non-interpretable decisions of pointwise classifications. Since the dynamics minimises the CLM grouping energy function and converges to local minima that correspond to locally optimal groupings, one can argue that a "locally optimal interpretation" is already included in the CLM evaluation. In consequence, however, it is not possible to judge the robustness of a single result. Only by performing several evaluations, the degree of variation of the obtained results can serve as a basis for such judgement.

In the evaluations of the previous sections, the standard deviation over four evaluation runs is therefore listed in the Tables 6.4 to 6.6. These tables reveal that the CLM evaluations

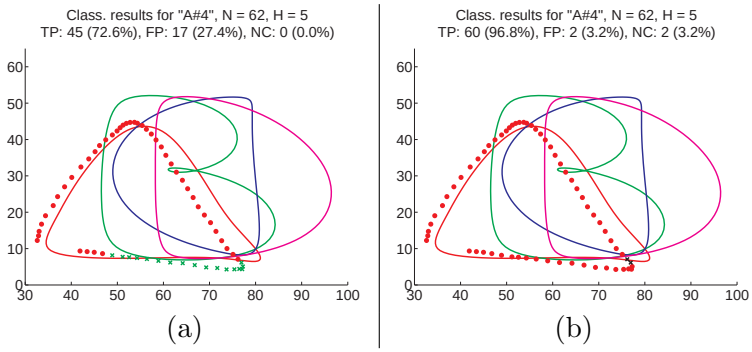



**Figure 6.17.:** In some cases, the classification results of the CLM are not robust. Figure (a-b) depict two different results produced on the same test case and with the same CLM parameters. The only difference in the evaluations were the different (random) initialisations of the neuron activities.

yield very robust results in most of the cases. In some cases, however, the evaluation results are not perfectly robust and yield standard deviations up to 12.1% (test sequence '$A_4$' for $N^{th} = 1, H = 5$ in Table 6.4). Figures 6.17(a-b) depict two corresponding results produced on the same test case and with the same CLM parameters, that yield different classification outcomes. The only difference in the evaluations were the different (random) initialisations of the neuron activities.

Since the characteristics of the test sequence do not necessarily provide evidence about the robustness of the evaluation, only multiple evaluations can show if the obtained results are robust or not.

### 6.3.9. On rejecting observed trajectories

Since the CLM dynamics converges to unique assignments of observation features corresponding to the available candidate motions, evaluations of the rejection behaviour of the combined CLM/UKR system cannot be made on the basis of the previous evaluation (which was the case for the UKR-only system).

In comparison to the UKR-only system, rejecting observations is a fundamentally different process with the CLM. Instead of applying a lower bound to compatibility values, a rejection of an observation corresponds to its assignment to the CLM background layer. The behaviour of this rejection mechanism can be influenced with the CLM background layer parameters $K_{bg}$ for the background neighbourhood radius and especially $\lambda_{bg}$ for the background strength.

For the evaluations in Section 6.3.5 to 6.3.7, these parameters have been optimised subject to three target conditions:

1. The system can reliably distinguish sequences corresponding to the trajectories in the validation data set (see Figure 6.1).

2. The background layer is weak enough such that the letter variations present in the validation set are not treated as background,

3. but the background is strong enough such that reversed validation trajectories are correctly rejected. These trajectories yield minimal compatibilities for the chronological order, but maximal compatibilities for the self-reconstruction. Therefore, they

| | $N^{th}=1$ | | | | $N^{th}=2$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1^R$ | 90.0 (6.7) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 70.0 (34.6) |
| $A_2^R$ | 84.1 (1.3) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 31.3 (0.0) |
| $A_3^R$ | 81.7 (2.4) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 23.3 (0.0) |
| $A_4^R$ | 77.0 (6.2) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 94.4 (11.3) | 38.7 (0.0) |
| $A_5^R$ | 86.0 (9.4) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 40.0 (0.0) | 24.0 (0.0) |
| $B_1^R$ | 89.7 (0.0) | 87.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_2^R$ | 89.8 (0.0) | 90.6 (6.3) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_3^R$ | 92.6 (4.9) | 87.3 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_4^R$ | 86.9 (0.0) | 86.9 (0.0) | 86.9 (0.0) | 86.9 (0.0) | 87.1 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_5^R$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_1^R$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 6.9 (0.0) |
| $C_2^R$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_3^R$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_4^R$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 30.0 (0.0) |
| $C_5^R$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_1^R$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_2^R$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 40.6 (0.0) |
| $D_3^R$ | 96.4 (7.1) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 9.4 (0.0) |
| $D_4^R$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 25.0 (0.0) |
| $D_5^R$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 11.6 (1.8) |

**Table 6.7.:** Overview of the classification results for the evaluations of *reversed* whole single letter sequences using the combined UKR/CLM system. In the case of the reversed sequences, true positive classifications correspond to rejections of the observations since the reversed letters are not represented in the candidate motions. General descriptions of the table structure and colours can be found in Table 6.4.

can be used to find a good compromise between both measures, $c^{loc}$ (6.14) and $c^{sre}$ (6.15).

The successful transfer of the first two conditions from the validation data set to the test data set (see Figure 6.1) are already covered by the evaluation details given in Tables 6.4 to 6.6. The successful transfer of the third condition can be seen in Table 6.7. The listed sequences correspond to the reversed trajectories from the first experiment (indicated by the superscript "$^R$" in the sequence names) described in Section 6.3.5 (Table 6.4). Since the sequences are reversed and therefore not represented by candidate motions in the CLM, true positive classifications correspond to correctly rejected observations. The table reveals that the rejection behaviour for the reversed trajectories works robustly in most of the cases. Poor classification results mostly occur for sparser sampled reversed '$A$' and '$D$' trajectories in combination with a large neighbourhood parameter (i.e. $N^{th}=2, H=20$). In addition, two of the reversed '$C$' sequences also yield decreased rejection performance for the same settings and '$A_5^R$', as single exception, also yields poor results for a shorter history, i.e. $N^{th}=2, H=15$.

In order to evaluate the rejection behaviour also for trajectories that are similar to the represented candidate motions, a new series of evaluations has been conducted with a CLM that only provides layers for the letters '$A$', '$B$', and '$D$' with an additional background layer. The '$C$' layer which was included in the previous evaluations has been excluded from the model. The corresponding test sequences '$C_1$' to '$C_5$' therefore constitute test sequences for non-represented trajectories. Since no corresponding candidate motion is represented in the CLM, correct classifications correspond to the rejection of the '$C$' observations.

| | $N^{th} = 1$ | | | | $N^{th} = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1$ | 90.7 (10.7) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_2$ | 79.8 (13.7) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_3$ | 80.0 (13.3) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_4$ | 72.6 (0.0) | 98.4 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_5$ | 68.0 (0.0) | 77.5 (15.0) | 100.0 (0.0) | 100.0 (0.0) | 68.0 (0.0) | 100.0 (0.0) | 70.0 (12.0) | 64.0 (0.0) |
| $B_1$ | 87.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 79.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_2$ | 85.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 95.5 (0.0) | 97.7 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_3$ | 91.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 93.8 (1.4) |
| $B_4$ | 94.3 (11.5) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_5$ | 64.1 (0.0) | 69.9 (3.8) | 69.2 (0.0) | 70.2 (0.6) | 66.7 (0.0) | 38.5 (0.0) | 51.3 (0.0) | 48.7 (0.0) |
| $C_1$ | 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) | 24.1 (0.0) |
| $C_2$ | 21.4 (0.0) | 0.0 (0.0) | 0.0 (0.0) | 0.0 (0.0) | 0.0 (0.0) | 22.9 (26.4) | 2.9 (0.0) | 32.9 (1.6) |
| $C_3$ | 8.9 (0.8) | 11.0 (3.2) | 0.0 (0.0) | 0.0 (0.0) | 8.1 (5.4) | 0.0 (0.0) | 0.0 (0.0) | 13.5 (0.0) |
| $C_4$ | 13.3 (0.0) | 0.0 (0.0) | 0.0 (0.0) | 0.0 (0.0) | 0.0 (0.0) | 0.0 (0.0) | 15.0 (11.1) | 46.7 (0.0) |
| $C_5$ | 0.0 (0.0) | 0.0 (0.0) | 0.0 (0.0) | 0.0 (0.0) | 0.0 (0.0) | 0.0 (0.0) | 16.7 (0.0) | 16.7 (0.0) |
| $D_1$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_5$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |

**Table 6.8.:** Overview of the classification results for the evaluation of single whole letter sequences. The underlying CLM consists of three layers for the letters '$A$', '$B$', and '$D$'. The fourth layer constitutes a background layer with parameters $\lambda_{bg} = (1 - 10/N)$ (for $N > 10$) and $K_{bg} = H$. These parameters have also been used for the evaluations described in Section 6.3.5 - 6.3.7. The letter '$C$' is not represented in the CLM. For the corresponding sequences '$C_1$' to '$C_5$', the listed true positive classifications therefore correspond to the rate of correctly rejected observations. Additional evaluations can be found in Table B.15 in Appendix B.2.4. General descriptions of the table structure and colours can be found in Table 6.4.

Table 6.8 gives an overview of the evaluations of the single whole letter trajectories using the CLM with excluded '$C$' layer. The parameter choice corresponds to the evaluations described in Section 6.3.5 to 6.3.7. In particular, the same settings for the background layer have been used: $\lambda_{bg} = (1 - 10/N)$ (for $N > 10$) and $K_{bg} = H$. The table reveals that the background layer is not strong enough to recognise the '$C$' trajectories as unknown and to reject them. Figure 6.18(a-c) depict three unsuccessful attempts to reject an unknown '$C$' trajectory. Since the background layer is configured rather weak, the CLM tends to assign the '$C$' observations to the class representations of '$B$' and '$D$', depending on the length of the considered history.

By increasing the strength of the background layer to $\lambda_{bg}^{strong} = 3\lambda_{bg}$, the CLM can be trimmed towards rejecting trajectories that are less similar to the candidate motions. Figure 6.18(d-f) show the results for the same sequences as in Figure 6.18(a-c), but evaluated with the stronger background. Whereas the results still are not perfect, they are much better than before.

This performance gain in rejecting unknown trajectories however comes with an undesirable side-effect. Table 6.9 lists the overview of the outcome from evaluations with the '$C$'-exclusive CLM using the stronger background layer ($\lambda_{bg}^{strong} = 3\lambda_{bg}$ and $K_{bg} = H$). Whereas the rejection results for the unknown '$C$' sequences indeed largely improved, the classification rates for the test sequences corresponding to the known candidate motions '$A$' and '$B$' largely suffer. This behaviour is logically consistent: the weak background layer

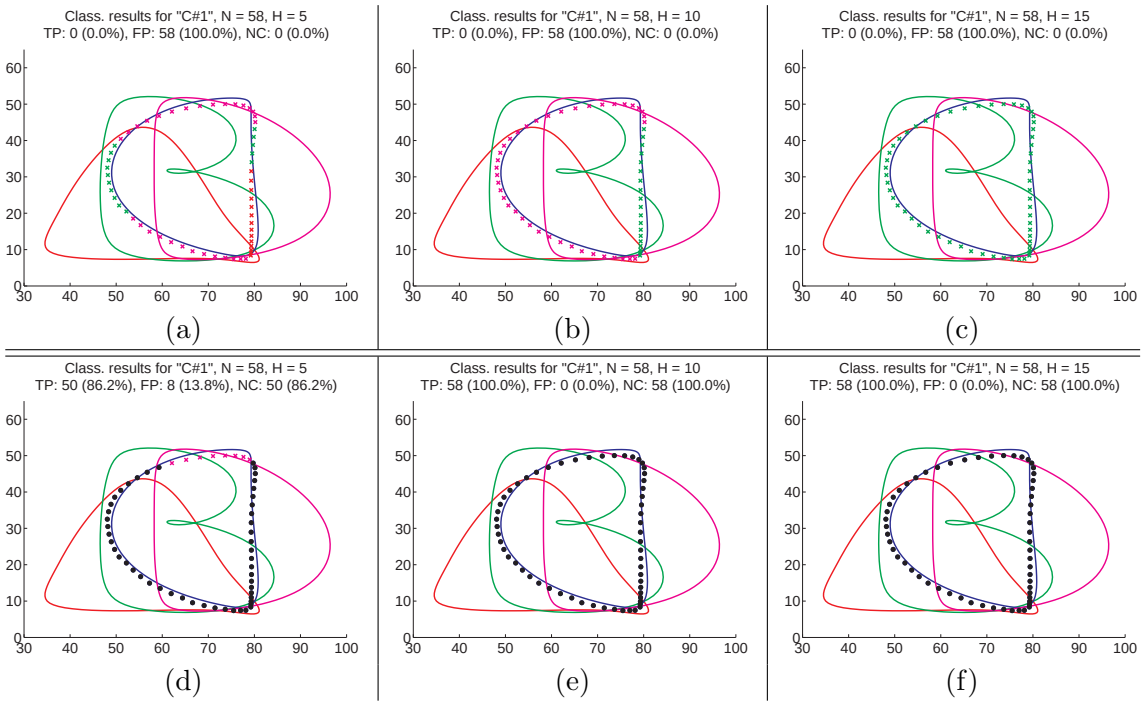



**Figure 6.18.:** Classification details for the unsuccessful '$C$' rejections with the '$C$'-exclusive CLM. **(a-c)** using a *weak* CLM background layer (i.e. $\lambda_{bg} = (1 - 10/N)$). **(d-f)** using a *strong* CLM background layer (i.e. $\lambda_{bg}^{strong} = 3\lambda_{bg}$). Black points depict true positive rejections. Depicted are reversed '$C_1$' trajectories for $N^{th} = 1$ and **(a,d)** $H = 5$, **(b,e)** $H = 10$, and **(c,f)** $H = 15$. General descriptions of the plot colours and data can be found in Figure 6.5.

used before implements the tendency to assign trajectories to one of the candidate motions instead of rejecting less compatible observations. In contrast to that, a strong background layer implements the contrary tendency and favours the rejection of trajectory parts over classifying a candidate motion on the basis of an uncertain decision.

The behaviour of the rejection mechanism therefore needs to be configured according to the requirements of the targeted task. If the set of possible sequences of observations is known beforehand and the occurrence of new motions is unlikely, a weak background is to be preferred. In contrast, if it is important that the observations precisely match the represented candidates and not represented motions can occur, a strong background is required.

### 6.3.10. Discussion

In this section, a combination of the previously described Structured UKR manifolds approach to motion recognition (see Section 6.2) and the recurrent neural Competitive Layer Model (CLM) has been presented. The basic principle has been already described in (Steffen et al., 2010). In this section, however, the evaluation of the method has largely been extended and an analysis of the CLM background layer and the related rejection behaviour has been added.

In correspondence to the UKR-only evaluation, three experiments have been conducted in order to evaluate the system performance on single whole letter trajectories, concatenated whole letter trajectories and concatenated parts of letter trajectories, respectively. In the

| | $N^{th} = 1$ | | | | $N^{th} = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1$ | 23.2 (17.2) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 67.9 (30.9) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_2$ | 40.9 (40.6) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_3$ | 49.2 (18.8) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_4$ | 16.5 (19.1) | 87.9 (17.7) | 100.0 (0.0) | 100.0 (0.0) | 84.7 (17.7) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $A_5$ | 46.5 (8.5) | 70.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 68.0 (0.0) | 100.0 (0.0) | 64.0 (0.0) | 64.0 (0.0) |
| $B_1$ | 19.6 (14.4) | 62.8 (11.2) | 80.8 (0.0) | 100.0 (0.0) | 23.7 (17.1) | 84.6 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_2$ | 34.1 (0.0) | 39.8 (0.0) | 73.0 (0.6) | 100.0 (0.0) | 27.3 (18.2) | 40.3 (1.1) | 59.7 (5.0) | 76.1 (27.6) |
| $B_3$ | 15.5 (18.6) | 66.9 (14.8) | 86.6 (15.5) | 100.0 (0.0) | 52.1 (12.1) | 88.2 (13.7) | 100.0 (0.0) | 85.4 (2.7) |
| $B_4$ | 51.6 (2.8) | 87.7 (14.2) | 100.0 (0.0) | 100.0 (0.0) | 87.1 (14.9) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_5$ | 37.5 (13.7) | 54.8 (10.9) | 38.5 (0.0) | 38.5 (0.0) | 35.3 (1.3) | 17.9 (20.8) | 0.0 (0.0) | 0.0 (0.0) |
| $C_1$ | 85.8 (2.9) | 100.0 (0.0) | 100.0 (0.0) | 0.0 (0.0) | 100.0 (0.0) | 98.3 (3.4) | 75.0 (50.0) | 100.0 (0.0) |
| $C_2$ | 90.7 (6.3) | 91.4 (17.1) | 53.6 (0.8) | 88.6 (22.9) | 92.9 (14.3) | 88.6 (22.9) | 100.0 (0.0) | 100.0 (0.0) |
| $C_3$ | 66.8 (14.0) | 45.2 (0.0) | 44.2 (0.7) | 43.5 (0.7) | 94.6 (10.8) | 87.2 (25.7) | 75.0 (29.1) | 100.0 (0.0) |
| $C_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_5$ | 100.0 (0.0) | 62.1 (25.6) | 44.6 (0.8) | 43.8 (0.8) | 74.2 (29.9) | 62.5 (25.4) | 76.7 (26.9) | 100.0 (0.0) |
| $D_1$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_5$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |

**Table 6.9.:** Overview of the classification results for the evaluation of single whole letter sequences. The underlying CLM consists of three layers for the letters '$A$', '$B$', and '$D$'. The fourth layer constitutes a background layer with parameters $\lambda_{bg}^{strong} = 3\lambda_{bg}$ and $K_{bg} = H$. The value of $\lambda_{bg}^{strong}$ is therefore three times higher than in the evaluations of Section 6.3.5 - 6.3.7. The letter '$C$' is not represented in the CLM. For the corresponding sequences '$C_1$' to '$C_5$', the listed true positive classifications therefore correspond to the rate of correctly rejected observations. Additional evaluations can be found in Table B.16 in Appendix B.2.5. General descriptions of the table structure and colours can be found in Table 6.4.

direct comparison with the evaluation outcomes of the UKR-only system, the combined system of UKR and CLM yields significantly improved results.

In additional experiments, the system's behaviour according to the rejection of unknown trajectory parts has been analysed in two contexts. In the first experiment, sequences with reversed trajectories of the known test letters have been evaluated with the same CLM structure utilised in the previous experiments (holding layers for all test letters '$A$', '$B$', '$C$', and '$D$' and an additional background layer). In the second experiment, in order to simulate the situation of unknown, but meaningful test sequences, a CLM only providing layers for the letters '$A$', '$B$', and '$D$' plus additional background layer has been evaluated. In particular, the classification behaviour for test sequences corresponding to the non-represented '$C$' letter have been analysed.

The experiments yield that a general recognition system needs to realise a compromise between the ability of robustly rejecting unknown observations on the one hand, and the ability to robustly recognise variations of known motions on the other hand.

Due to the intuitive parametrisation of the CLM background layer and the corresponding rejection behaviour, such compromise can be found straightforwardly. In addition, task-related requirements for strong or weak rejection behaviour (or weak or strong recognition behaviour of candidate motions, respectively) can also be achieved by intuitively configuring the background layer to be strong or weak.

In contrast to the UKR-only approach, the combined system is not inherently qualified for an online application. Due to the inclusion of the CLM dynamics, the classification of a single observation can be affected in principle by all considered observations in the evaluated sequence. Whereas it is indeed possible to apply the method to newly perceived data streams, the dynamics largely benefits from a more holistic view on the observation sequence. Moreover, the need for convergence of the dynamics is time consuming and therefore a limiting factor. An iterative approach to accelerate the convergence for online applications is possible.

In connection to this idea, initial experiments have been conducted in which the processed observation sequence has been iteratively extended. In each step, the CLM dynamics has been applied until convergence. In this scenario, the neurons corresponding to the observations already processed in the previous iteration can be initialised with assignments that are non-unique, but nevertheless biased towards the previous classification outcome. Such procedure largely accelerates the convergence if the newly perceived observations belong to the same class as the previously classified ones.

Whereas in this situation, a certain momentum effect (like it can be observed with the UKR-only approach for concatenated sequences) can be expected, the system is able to propagate later occurrences of disambiguating events in the observation sequence back to earlier observations due to the dynamics. The CLM mechanism therefore enables a later correction of previously recognition results.

In contrast to the UKR-only approach, the classification results from the UKR/CLM combination yield always unique assignments to one class and do therefore not leave much space for interpretations. Due to the convergence to unique class assignments, a single evaluation yields no insights about the robustness of the result or about similarities of different trajectory parts with multiple candidate motions. Whereas such analysis is basically possible with the UKR-only system, it also usually means that a corresponding interpretation may be necessary in order to decide about the final UKR-only results.

From this viewpoint, the CLM outcome constitutes on the one hand a decrease of possibilities for further processing or interpretation, but also provides on the other hand a more sophisticated result, that corresponds to a (locally) optimal solution of a feature segmentation mechanism and in fact also needs less post-processing.

The analysis of the rejection behaviour reveals that the combined UKR/CLM system is better suited for the rejection of unknown motions than the UKR-only approach. By means of the CLM background layer parameters $\lambda_{bg}$ and $K_{bg}$, the rejection behaviour can be intuitively configured. In addition, also sparsely sampled trajectories can be handled robustly. Due to the CLM dynamics in combination with the symmetric neighbourhood consideration for the CLM features, the momentum effect is not present in the form observed with the UKR-only system.

To summarise, the combined UKR/CLM system share the following advantages with the previous UKR-only approach:

- Due to the definition of the compatibility features in the manifold domain instead of directly in the observation space, it is independent – to a certain extent – of the specific task or observation space characteristics.
- It can handle various sampling rates of the observations.

- It does not require a fixed number of observations and can therefore be applied on the 'raw' data stream without specific preprocessing in order to interpolate to specific sampling rates or observation counts.
- It does not require the observations of a complete motion, but is also able to classify and segment trajectory parts.
- The ability to recognise a specific motion is automatically included in its representation as a Structured UKR manifold for reproduction and synthesis, because the recognition mechanism directly works on the basis of these reproduction manifolds.

In addition, the following list of advantages can be added:

- The classification outcome corresponds to a (local) minimum of a grouping energy function.
- The classification rates significantly outperform the results of the UKR-only approach.
- The CLM background layer realises an improved rejection behaviour for unknown motions.
- The parameters of this background layer can be intuitively configured.
- Due to the inclusion of the CLM dynamics, later occurrences of trajectory characteristics in the processed sequence can be propagated back to previous observations. Such back-propagation was not possible in the UKR-only approach.
- As one consequence of the previous point, the momentum effect observed in the UKR-only approach is largely decreased.
- The improved self-reconstruction compatibility measure produces more robust classification features.

Given this list of convincing additional advantages, the lack of post-classification interpretability is a minor issue.

At this point, it shall be mentioned again that, whereas the evaluation is carried out for the example of the letter data set, the analysed issues constitute general aspects of the method. Since the compatibilities along with the history consideration are defined in terms of the manifolds, the transitions between classes or candidate motions occur for all kinds of data that is represented in Structured UKR manifolds in the same way.

As a final remark concerning the computational demand of the approach, it shall be mentioned that, in addition to the computations required for the UKR-only approach, the CLM requires the convergence of its dynamics. This convergence is guaranteed under certain weak conditions (see Section 6.3.1), but the amount of required steps of the iterative algorithm can largely differ for different sequences. For the sequences used to evaluate the system, convergence could be encountered in a range from only 10 complete iterations of the algorithm for short and distinct sequences up to 300 complete iterations for long and/or intricate sequences. Whereas 10 iterations do not yield crucial time costs, the amount of 300 iterations requires a non-negligible time. In addition to the time needed to compute the input features (which is also required by the UKR-only approach), the CLM does not

constitute a system that can operate in real time. Since it is usually not necessary to run the whole classification or segmentation mechanism for every new observation, but only after a certain amount of new observations, the system may still be used in an online scenario. A sparser trajectory sampling can in addition speed up the evaluations.

# 7. Conclusions

In Chapter 3, a new method to acquire hand posture data for anthropomorphic robotic hands with a data glove was presented. The developed mapping is based on an explicit model of the glove sensor dependencies and provides model solutions for the abduction sensors and the thumb base sensors. In both cases, this required to overcome shortcomings of the glove design.

The mapping from glove to robotic hand was implemented in two stages. The first stage maps the raw sensor values onto a joint angle representation close to the human hand (and can therefore be seen as a kinematic model of the glove). The second stage is a specialisation for the ShadowHand. This two-part approach does not only allow to express the recorded hand postures in terms of the human hand model, but also constitutes a basis for further specialisations to other (robotic) hand models. Along with the mapping from the human hand model to the ShadowHand, an analytic way of computing the nonintuitive ShadowHand thumb configuration on the basis of the simpler thumb roll/abduction representation was presented.

The implementation of the mapping provided the basis for the acquisition of anthropomorphic hand posture data. Furthermore, using the developed mapping, the CyberGlove II still plays an important role in the acquisition of hand posture data in the Neuroinformatics Group at the University of Bielefeld, even though a sophisticated vision system is also available now.

In Chapter 4, an experience-based approach to dextrous grasping was presented. By representing the underlying grasping experience in the form of a manifold of hand postures, the method gave rise to the idea of Structured Manifolds, which constitute the basis for the remainder of the thesis. Further analyses of the characteristics of the Grasp Manifolds provided insights not only concerning the grasping data, but also about important issues concerning the subsequent development of Manipulation Manifolds.

Chapter 5 focussed on the generation of Manipulation Manifolds as an example for Structured Manifolds. Here, two different methods were presented for the exemplary manipulation of turning a bottle cap. The first one iteratively constructs the manifold from a classified set of training sequences and the second one uses a learning approach for the generation of the manifold. To this end, three extensions to UKR were presented that enable the method to represent periodic latent spaces, to incorporate prior knowledge about sequence affiliations and to propagate the chronological order of training observations to their latent representations. The analysis of the presented learning approach, on an appropriate toy data set, provided guidelines for the handling of the new parameters, which could be successfully applied to the learning of the task of turning a bottle cap.

Subsequently, three different ways of reproducing the represented manipulations have been described, from a simple open-loop navigation through latent space to more sophisticated closed-loop control. The latter has been successfully implemented on a real robotic hand, namely the Gifu Hand III. By learning a UKR representation of the manipulation

of swapping Chinese health balls using hand posture information in connection with the corresponding ball states, the closed-loop control was able to produce a more sophisticated manipulation behaviour.

Finally, Chapter 6 addressed the use of Structured Manifolds for the purpose of recognition and segmentation. To this end, a set of candidate manifolds, each representing a different motion trajectory, was initially trained and used as prototype motions. For newly observed sequences, features in the manifold domain of the candidate motions were calculated that exploited the clear and previously known structure of the manifolds. These features were used to rate the level of compatibility of the observed and represented motions.

In the first approach, these compatibilities are directly compared with each other and a winner-take-all mechanism associates the observations with the candidate motion that yields the highest compatibility values.

In addition, a second approach was presented, which extends the initial manifold-only approach by neural competition of the candidate motions using the CLM. The compatibility features, which were fed directly into the classification model of the manifold-only approach using a winner-take-all mechanism, were then used as excitatory and inhibitory interactions between features in the neural grouping energy of the CLM. Applying the CLM dynamics to minimise this energy results in locally optimal groupings of the features and therefore of the underlying observations, respectively.

## 7.1. Outlook and Future Work

The work presented in this thesis consists of various aspects each of which may be extended in a number of directions.

**Grasping and Grasp Manifolds** Grasp Manifolds were the basis for the idea of the Structured Manifolds that were introduced in Chapter 5. However, the grasping representation itself can give rise to interesting directions for future work.

In particular, the benefits of using manifold approaches other than SOMs, such as UKR, GPLVM and others, could be analysed in order to improve the accuracy of the representation and the grasping performance.

In addition, a more elaborate combination of Grasp and Manipulation Manifolds may be developed in which a specific sub-region of the Manipulation Manifold constitutes the grasp experience and can be used to grasp an object in order to initialise a subsequent manipulation. More generally, the whole Grasp Manifold may not need to be part of the Manipulation Manifold. In the example of turning a bottle cap presented in Chapter 5, a Grasp Manifold could represent a wide range of bottle cap grasp postures of the hand, and only a small subset may also be part of the manipulation movement. Such a Grasp Manifold could then be used to grasp the bottle cap and a navigation through the Grasp Manifold towards the intersection of Grasp and Manipulation Manifold could lead to an initialisation of the manipulation movement.

**Manipulation Manifolds** Structured Manifolds were introduced for dextrous manipulation using UKR. In this task-related scenario, a variety of future directions can be taken.

Firstly, the performance of other manifold learning methods could be evaluated. In particular, the use of GPLVM, which has lately received a lot of attention in the world of

robotics, machine learning and computer animation, is a possible alternative to UKR and would at the same time produce additional interest in the community due to the popularity of the GPLVM itself.

In addition, some interesting issues arising from the manifold generation procedure remain for future work. In particular, the way of incorporating more than one motion parameter (like the radius of the cap in the cap turning example), especially in the case of the automatic learning approach, requires further investigation.

Furthermore, manifold generation in Chapter 5 always considered the whole training sequence when optimising the UKR manifold. In scenarios like turning the bottle cap, it may be beneficial to represent the whole movement, but only adapt the manifold to the manipulation parts, i.e., those parts of the sequences in which the fingers have contact with the bottle cap. Since only these parts characterise the cap radius, they exclusively hold the important information about the motion parameter, which is the radius of the cap.

**Manipulation** For the actuation of the represented manipulation movements, three different control schemes were presented, one of which was implemented on the real robotic Gifu Hand III for the task of swapping Chinese health balls. The manipulation of turning a bottle cap was only demonstrated in simulation. The transfer of the task to a real robotic hand remains future work due to the lack of appropriate hardware. The Gifu Hand III is actuated by servomotors in the finger joints and is therefore qualified to move accurately and fast to a desired hand posture. Tactile sensors (or alternatively compliant joints), which are necessary for the manipulation, are however not available in the hand. In contrast, the ShadowHand has recently been equipped with adequate tactile sensors, but the actuation controller in combination with the hardware does not produce the required accuracy and smoothness in the motion generation. It is expected that soon hardware will be able to perform such control, at which point the simulated work presented here can be applied to a physical robot, as it has already been done with the ball swapping manipulation on the Gifu Hand II.

**Structured Manifolds for Motion Recognition** The use of Structured Manifolds for motion recognition and segmentation was analysed in a letter trajectories scenario. Future work may transfer the method to other sorts of data trajectories, in particular its application to manipulation data from the CyberGlove will be addresses in the near future. To this end, it is planned to represent a set of candidate manipulation motions (e.g, cap turning) as Structured Manifolds and extending this to other motions such as using scissors and writing with a pen or on a keyboard. By including the position of the hand, recorded by a vision system, more complex movements such as knotting, juggling and pouring can be analysed with respect to the recognition and segmentation performance of the system.

**A Combined System for Motion Production and Recognition** One of the most interesting fields for future work may be the combination of the two systems into one system that is able to produce and recognise a set of candidate motions at the same time. Such a combination could be applied to the case of manipulation, where the manifolds only hold the hand postures as was presented in the case of turning the bottle cap. Due to the flexibility of the UKR approach, however, even more complex scenarios are conceivable. For instance, the motions of two arm/hand pairs in bimanual tasks could be represented in one manifold. If one arm/hand pair (or even a demonstrator) performs one of the candidate motions, the

*associative completion* mechanism (used in the closed-loop controller described in Section 5.6 to project partial hand/ball observations onto the manifold) could be used to recognise the correct candidate and find an adequate completion, i.e. the posture of the second arm/hand pair. Subsequently, a closed-loop control similar to the one used for swapping the Chinese health balls (see Section 5.6) could be applied in order to finish the bimanual task, either in the scenario of two robotic arms or with one robotic arm in cooperation with the human demonstrator.

**Structured Manifolds in general** The general case of Structured Manifolds yields a variety of possible application domains. Since the formulation of UKR and its new extensions are basically suitable for a wide range of data sequences, the consideration of other domains apart from dextrous manipulation or even motion representation, has the possibility to become an interesting future topic.

# A. CyberGlove II

## A.1. From $\alpha/\beta$- to $\Theta_0/\Theta_1$-Representation

To get the mapping from the $\alpha/\beta$- to the $\Theta_0/\Theta_1$-representation and back (see Section 3.5), the position of a point $\mathbf{p}$ on the thumb axis with unit distance to the thumb base frame is calculated.

The coordinates of $\mathbf{p}$ in $\alpha/\beta$-representation can be obtained straightforwardly by using trigonometric functions as visualised in Figure A.1(a):

$$\mathbf{p}^{\alpha/\beta} = \begin{pmatrix} \sin(\alpha)\cos(\beta) \\ \cos(\alpha) \\ \sin(\alpha)\sin(\beta) \end{pmatrix}. \tag{A.1}$$

For the $\Theta_0/\Theta_1$-representation, Homogeneous Transformations are utilised. To this end – as shown in Figure A.1(b) – first, the $x$-axis of the thumb base frame is rotated onto the TH0 joint axis by rotating about 45° around the $z$-axis (transformation $\mathbf{T}_0$). Afterwards, the transformations for the thumb joints apply. Here, the transformation for joint TH0 ($\mathbf{T}_{TH0}$) corresponds to a rotation around the new $x'$-axis and the transformation for joint TH1 ($\mathbf{T}_{TH1}$) afterwards corresponds to a rotation around the new $y''$-axis:

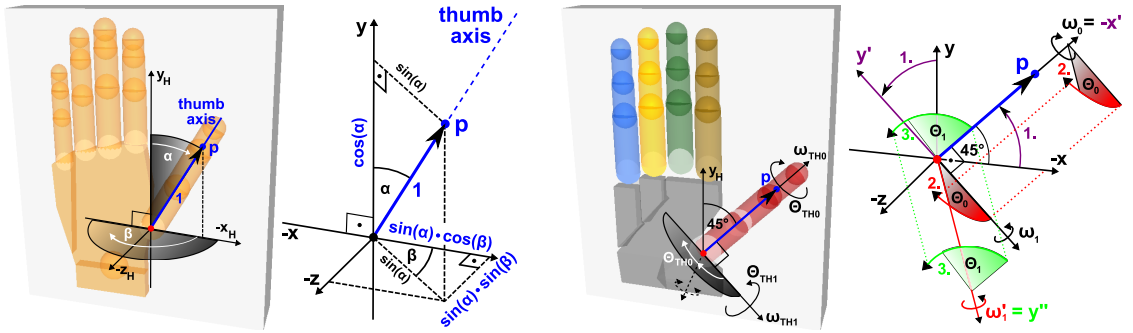
$$\begin{aligned} \mathbf{T} &= \mathbf{T}_0\ \mathbf{T}_{TH0}\ \mathbf{T}_{TH1} && \text{(A.2)} \\ &= \mathbf{R}_{z,45^\circ}\ \mathbf{R}_{x',\Theta_0}\ \mathbf{R}_{y'',-\Theta_1} && \text{(A.3)} \end{aligned}$$

whereas $\mathbf{R}_{x,\phi}$, $\mathbf{R}_{y,\phi}$ and $\mathbf{R}_{z,\phi}$ are the standard Homogeneous Transformations for rotations about the angle $\phi$ around the $x$-, $y$- and $z$-axes, respectively. The coordinates of point $\mathbf{p}$ in $\Theta_0/\Theta_1$-representation then corresponds to the normalised $x$-axis of $\mathbf{T}$ which resolves to:

$$\mathbf{p}^{\Theta_0/\Theta_1} = \begin{pmatrix} \frac{1}{2}\sqrt{2}\left[\cos(\Theta_1) + \sin(\Theta_0)\sin(\Theta_1)\right] \\ \frac{1}{2}\sqrt{2}\left[\cos(\Theta_1) - \sin(\Theta_0)\sin(\Theta_1)\right] \\ \cos(\Theta_0)\sin(\Theta_1). \end{pmatrix} \tag{A.4}$$

Equating the $\alpha/\beta$- and $\Theta_0/\Theta_1$-representation for point $\mathbf{p}$ then yields $\mathbf{p}^{\Theta_0/\Theta_1} \stackrel{!}{=} \mathbf{p}^{\alpha/\beta}$ and thus the following system of equations:

$$\begin{pmatrix} \frac{1}{2}\sqrt{2}\left[\cos(\Theta_1) + \sin(\Theta_0)\sin(\Theta_1)\right] \\ \frac{1}{2}\sqrt{2}\left[\cos(\Theta_1) - \sin(\Theta_0)\sin(\Theta_1)\right] \\ \cos(\Theta_0)\sin(\Theta_1) \end{pmatrix} \stackrel{!}{=} \begin{pmatrix} \sin(\alpha)\cos(\beta) \\ \cos(\alpha) \\ \sin(\alpha)\sin(\beta) \end{pmatrix} \tag{A.5}$$




(a) Describing the coordinates of point **p** in dependence of $\alpha$ and $\beta$: **p** can straightforwardly be denoted in terms of sin and cos.

(b) Describing **p** in terms of the thumb joint angles $\Theta_0$ and $\Theta_1$: first, the $x$-axis is rotated onto the TH0-axis (step "1.", violet), then the rotations around joint axes TH0 about $\Theta_0$ (step "2.", red) and TH1 about $\Theta_1$ (step "3.", green) apply.

**Figure A.1.:** The two different representations of point **p** (blue) as a basis for the mapping from $\alpha/\beta$- to $\Theta_0/\Theta_1$-representation and back.

Solving this system for $\Theta_0/\Theta_1$ yields:

$$\Theta_0(\alpha,\beta) = \arctan 2\left(\frac{1}{2}\frac{\sqrt{2}(-\cos(\alpha)+\sin(\alpha)\cos(\beta))}{\sqrt{\frac{1}{2}-\cos(\alpha)\sin(\alpha)\cos(\beta)+\frac{1}{2}\sin(\alpha)^2\sin(\beta)^2}}, \frac{\sin(\alpha)\sin(\beta)}{\sqrt{\frac{1}{2}-\cos(\alpha)\sin(\alpha)\cos(\beta)+\frac{1}{2}\sin(\alpha)^2\sin(\beta)^2}}\right). \tag{A.6}$$

$$\Theta_1(\alpha,\beta) = \arctan 2\left(\sqrt{\frac{1}{2}-\cos(\alpha)\sin(\alpha)\cos(\beta)+\frac{1}{2}\sin(\alpha)^2\sin(\beta)^2}, \frac{1}{2}\sqrt{2}(\cos(\alpha)+\sin(\alpha)\cos(\beta))\right). \tag{A.7}$$

In the inverse direction, solving the system for $\alpha/\beta$ yields:

$$\alpha(\Theta_0,\Theta_1) = \arctan 2\left(\sqrt{\frac{1}{2}+\cos(\Theta_1)\sin(\Theta_0)\sin(\Theta_1)+\frac{1}{2}\sin(\Theta_1)^2\cos(\Theta_0)^2}, \frac{1}{2}\sqrt{2}\left(\cos(\Theta_1)-\sin(\Theta_1)\sin(\Theta_0)\right)\right). \tag{A.8}$$

$$\beta(\Theta_0,\Theta_1) = \arctan 2\left(\frac{\cos(\Theta_0)\sin(\Theta_1)}{\sqrt{\frac{1}{2}+\cos(\Theta_1)\sin(\Theta_0)\sin(\Theta_1)+\frac{1}{2}\sin(\Theta_1)^2\cos(\Theta_0)^2}}, \frac{1}{2}\frac{\sqrt{2}\left(\cos(\Theta_1)+\sin(\Theta_1)\sin(\Theta_0)\right)}{\sqrt{\frac{1}{2}+\cos(\Theta_1)\sin(\Theta_0)\sin(\Theta_1)+\frac{1}{2}\sin(\Theta_1)^2\cos(\Theta_0)^2}}\right) \tag{A.9}$$

A visualisation of these mappings can be found in section 3.5 (Figure 3.11).

# B. Supplemental evaluation results for the motion recognition approach

## B.1. Recognition with the UKR-only system

### B.1.1. Single whole letter trajectories ($N^{th} = 3, 4$)

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1$ | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | n/a |
| $A_2$ | 95.2 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | n/a |
| $A_3$ | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | n/a |
| $A_4$ | 100.0 | 100.0 | 100.0 | 100.0 | 93.8 | 100.0 | 100.0 | n/a |
| $A_5$ | 94.1 | 100.0 | 100.0 | n/a | 76.9 | 100.0 | n/a | n/a |
| $B_1$ | 80.8 | 80.8 | 92.3 | 92.3 | 90.0 | 90.0 | 90.0 | 90.0 |
| $B_2$ | 93.3 | 93.3 | 96.7 | 96.7 | 86.4 | 95.5 | 95.5 | 95.5 |
| $B_3$ | 91.7 | 91.7 | 91.7 | 91.7 | 83.3 | 83.3 | 83.3 | n/a |
| $B_4$ | 90.5 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | n/a |
| $B_5$ | 80.8 | 84.6 | 92.3 | 92.3 | 90.0 | 90.0 | 90.0 | 90.0 |
| $C_1$ | 85.0 | 85.0 | 85.0 | 85.0 | 86.7 | 86.7 | 86.7 | n/a |
| $C_2$ | 75.0 | 87.5 | 87.5 | 87.5 | 77.8 | 88.9 | 88.9 | n/a |
| $C_3$ | 60.0 | 68.0 | 60.0 | 60.0 | 57.9 | 57.9 | 57.9 | n/a |
| $C_4$ | 75.0 | 85.0 | 85.0 | 85.0 | 93.3 | 93.3 | 93.3 | n/a |
| $C_5$ | 55.0 | 55.0 | 55.0 | 55.0 | 60.0 | 60.0 | 66.7 | n/a |
| $D_1$ | 100.0 | 100.0 | 100.0 | 100.0 | 94.1 | 94.1 | 94.1 | n/a |
| $D_2$ | 95.2 | 95.2 | 95.2 | 95.2 | 93.8 | 93.8 | 93.8 | n/a |
| $D_3$ | 100.0 | 100.0 | 100.0 | 100.0 | 93.8 | 93.8 | 93.8 | n/a |
| $D_4$ | 100.0 | 100.0 | 100.0 | n/a | 92.9 | 92.9 | n/a | n/a |
| $D_5$ | 100.0 | 100.0 | 100.0 | n/a | 100.0 | 100.0 | n/a | n/a |

**Table B.1.:** List of evaluation results for single whole letter trajectories using the UKR-only system for $N^{th} = 3, 4$. This table constitutes the extension of Table 6.1 in Section 6.2.2.

### B.1.2. Two concatenated whole letter trajectories

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1B_1$ | 86.0 | 86.0 | 92.0 | 86.0 | 92.1 | 92.1 | 86.8 | 76.3 |
| $A_1C_1$ | 90.9 | 79.5 | 72.7 | 56.8 | 84.8 | 75.8 | 60.6 | 54.5 |
| $A_1D_1$ | 93.6 | 89.4 | 78.7 | 78.7 | 88.6 | 85.7 | 80.0 | 74.3 |
| $A_2B_2$ | 86.3 | 88.2 | 88.2 | 86.3 | 84.2 | 89.5 | 86.8 | 78.9 |
| $A_2C_2$ | 82.2 | 77.8 | 71.1 | 64.4 | 79.4 | 70.6 | 67.6 | 52.9 |
| $A_2D_2$ | 90.5 | 92.9 | 78.6 | 73.8 | 90.6 | 90.6 | 75.0 | 68.8 |
| $A_3B_3$ | 90.9 | 90.9 | 86.4 | 81.8 | 93.9 | 84.8 | 81.8 | 72.7 |
| $A_3C_3$ | 75.6 | 75.6 | 66.7 | 66.7 | 73.5 | 70.6 | 67.6 | 58.8 |
| $A_3D_3$ | 92.7 | 92.7 | 82.9 | 73.2 | 96.8 | 87.1 | 74.2 | 71.0 |
| $A_4B_4$ | 95.2 | 97.6 | 88.1 | 85.7 | 96.9 | 93.8 | 87.5 | 78.1 |
| $A_4C_4$ | 85.4 | 80.5 | 70.7 | 63.4 | 80.6 | 71.0 | 58.1 | 51.6 |
| $A_4D_4$ | 92.5 | 92.5 | 80.0 | 75.0 | 90.0 | 93.3 | 73.3 | 70.0 |
| $A_5B_5$ | 86.0 | 88.4 | 88.4 | 88.4 | 87.9 | 93.9 | 93.9 | 93.9 |
| $A_5C_5$ | 64.9 | 67.6 | 67.6 | 51.4 | 57.1 | 64.3 | 57.1 | 46.4 |
| $A_5D_5$ | 94.4 | 94.4 | 80.6 | 72.2 | 85.2 | 81.5 | 70.4 | 51.9 |

**Table B.2.:** List of evaluation results for two concatenated whole letter trajectories using the UKR-only system for $N^{th} = 3, 4$. This table constitutes the extension of Table 6.2 in Section 6.2.3.

| | $N^{th} = 1$ | | | | $N^{th} = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $B_1A_1$ | 69.6 | 85.8 | 91.2 | 91.2 | 89.2 | 97.3 | 97.3 | 97.3 |
| $B_1C_1$ | 68.4 | 79.4 | 81.6 | 75.7 | 83.8 | 83.8 | 80.9 | 73.5 |
| $B_1D_1$ | 72.6 | 85.6 | 89.7 | 89.7 | 89.0 | 95.9 | 91.8 | 83.6 |
| $B_2A_2$ | 74.8 | 85.4 | 93.4 | 93.4 | 89.5 | 94.7 | 94.7 | 94.7 |
| $B_2C_2$ | 65.8 | 75.9 | 81.0 | 77.2 | 78.5 | 77.2 | 79.7 | 73.4 |
| $B_2D_2$ | 77.5 | 85.4 | 91.4 | 90.7 | 89.5 | 92.1 | 88.2 | 82.9 |
| $B_3A_3$ | 79.4 | 92.4 | 89.3 | 90.8 | 89.4 | 89.4 | 87.9 | 90.9 |
| $B_3C_3$ | 61.1 | 75.0 | 68.1 | 68.1 | 72.6 | 65.8 | 68.5 | 67.1 |
| $B_3D_3$ | 80.6 | 91.8 | 88.1 | 86.6 | 88.2 | 85.3 | 77.9 | 76.5 |
| $B_4A_4$ | 80.5 | 94.3 | 94.3 | 93.5 | 95.2 | 93.5 | 98.4 | 98.4 |
| $B_4C_4$ | 79.3 | 86.0 | 87.6 | 90.9 | 83.6 | 88.5 | 93.4 | 82.0 |
| $B_4D_4$ | 81.9 | 94.0 | 92.2 | 87.9 | 93.2 | 86.4 | 86.4 | 74.6 |
| $B_5A_5$ | 73.4 | 82.0 | 85.9 | 89.1 | 78.1 | 90.6 | 92.2 | 95.3 |
| $B_5C_5$ | 65.2 | 65.9 | 62.3 | 63.8 | 60.9 | 65.2 | 69.6 | 66.7 |
| $B_5D_5$ | 77.4 | 83.5 | 84.2 | 85.0 | 80.6 | 88.1 | 83.6 | 80.6 |
| $C_1A_1$ | 84.4 | 93.8 | 96.1 | 96.1 | 92.2 | 95.3 | 95.3 | 84.4 |
| $C_1B_1$ | 68.4 | 79.4 | 83.8 | 87.5 | 86.8 | 95.6 | 95.6 | 95.6 |
| $C_1D_1$ | 90.5 | 92.1 | 90.5 | 88.9 | 90.5 | 87.3 | 77.8 | 76.2 |
| $C_2A_2$ | 72.9 | 82.7 | 87.2 | 89.5 | 82.1 | 86.6 | 85.1 | 77.6 |
| $C_2B_2$ | 65.8 | 77.8 | 86.1 | 86.1 | 81.0 | 86.1 | 92.4 | 92.4 |
| $C_2D_2$ | 76.7 | 82.0 | 80.5 | 80.5 | 80.6 | 77.6 | 79.1 | 76.1 |
| $C_3A_3$ | 70.7 | 80.5 | 79.7 | 82.0 | 79.1 | 77.6 | 73.1 | 64.2 |
| $C_3B_3$ | 61.8 | 79.2 | 74.3 | 79.2 | 76.7 | 74.0 | 78.1 | 82.2 |
| $C_3D_3$ | 72.1 | 76.5 | 73.5 | 72.8 | 76.8 | 68.1 | 66.7 | 65.2 |
| $C_4A_4$ | 80.3 | 86.9 | 89.3 | 95.1 | 88.5 | 96.7 | 90.2 | 83.6 |
| $C_4B_4$ | 80.2 | 89.3 | 90.1 | 91.7 | 86.9 | 90.2 | 96.7 | 96.7 |
| $C_4D_4$ | 80.0 | 82.6 | 85.2 | 86.1 | 82.8 | 86.2 | 77.6 | 75.9 |
| $C_5A_5$ | 69.1 | 73.6 | 75.5 | 79.1 | 69.1 | 78.2 | 69.1 | 58.2 |
| $C_5B_5$ | 68.8 | 74.6 | 76.1 | 77.5 | 68.1 | 78.3 | 81.2 | 81.2 |
| $C_5D_5$ | 75.7 | 74.8 | 71.3 | 68.7 | 70.7 | 67.2 | 62.1 | 56.9 |
| $D_1A_1$ | 89.9 | 97.8 | 100.0 | 100.0 | 97.1 | 100.0 | 100.0 | 95.7 |
| $D_1B_1$ | 74.7 | 84.2 | 87.0 | 89.7 | 91.8 | 98.6 | 100.0 | 100.0 |
| $D_1C_1$ | 90.5 | 92.1 | 90.5 | 91.3 | 92.1 | 90.5 | 84.1 | 77.8 |
| $D_2A_2$ | 88.1 | 94.4 | 97.6 | 98.4 | 93.8 | 96.9 | 92.2 | 82.8 |
| $D_2B_2$ | 78.1 | 87.4 | 94.7 | 92.7 | 89.5 | 93.4 | 96.1 | 97.4 |
| $D_2C_2$ | 76.7 | 82.7 | 83.5 | 85.0 | 80.6 | 82.1 | 79.1 | 76.1 |
| $D_3A_3$ | 94.3 | 99.2 | 100.0 | 100.0 | 98.4 | 100.0 | 100.0 | 88.7 |
| $D_3B_3$ | 82.1 | 94.0 | 92.5 | 94.0 | 91.2 | 92.6 | 94.1 | 100.0 |
| $D_3C_3$ | 73.5 | 80.9 | 76.5 | 77.9 | 79.7 | 75.4 | 76.8 | 71.0 |
| $D_4A_4$ | 82.1 | 93.2 | 96.6 | 98.3 | 96.6 | 100.0 | 96.6 | 88.1 |
| $D_4B_4$ | 81.9 | 95.7 | 97.4 | 94.0 | 94.9 | 93.2 | 100.0 | 100.0 |
| $D_4C_4$ | 80.9 | 85.2 | 87.8 | 89.6 | 86.2 | 89.7 | 82.8 | 74.1 |
| $D_5A_5$ | 85.7 | 93.3 | 97.1 | 99.0 | 92.5 | 100.0 | 96.2 | 83.0 |
| $D_5B_5$ | 81.2 | 90.2 | 93.2 | 94.0 | 85.1 | 95.5 | 97.0 | 100.0 |
| $D_5C_5$ | 76.5 | 79.1 | 73.9 | 70.4 | 75.9 | 70.7 | 70.7 | 67.2 |

**Table B.3.:** List of evaluation results for two concatenated whole letter trajectories using the UKR-only system for $N^{th} = 1, 2$. This table constitutes the extension of Table 6.2 in Section 6.2.3.

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $B_1A_1$ | 90.0 | 90.0 | 96.0 | 90.0 | 94.7 | 94.7 | 86.8 | 78.9 |
| $B_1C_1$ | 80.4 | 73.9 | 78.3 | 63.0 | 82.9 | 71.4 | 65.7 | 51.4 |
| $B_1D_1$ | 89.8 | 83.7 | 79.6 | 71.4 | 91.9 | 83.8 | 70.3 | 67.6 |
| $B_2A_2$ | 94.1 | 96.1 | 98.0 | 90.2 | 92.1 | 97.4 | 92.1 | 84.2 |
| $B_2C_2$ | 83.3 | 77.8 | 74.1 | 68.5 | 75.0 | 75.0 | 70.0 | 70.0 |
| $B_2D_2$ | 90.2 | 90.2 | 82.4 | 74.5 | 86.8 | 84.2 | 71.1 | 65.8 |
| $B_3A_3$ | 95.5 | 95.5 | 95.5 | 86.4 | 90.9 | 90.9 | 81.8 | 69.7 |
| $B_3C_3$ | 73.5 | 75.5 | 67.3 | 67.3 | 67.6 | 62.2 | 62.2 | 56.8 |
| $B_3D_3$ | 93.3 | 84.4 | 73.3 | 68.9 | 88.2 | 79.4 | 67.6 | 61.8 |
| $B_4A_4$ | 95.2 | 100.0 | 100.0 | 90.5 | 96.9 | 96.9 | 84.4 | 68.8 |
| $B_4C_4$ | 82.9 | 92.7 | 78.0 | 70.7 | 93.5 | 80.6 | 71.0 | 51.6 |
| $B_4D_4$ | 92.5 | 87.5 | 75.0 | 72.5 | 93.3 | 76.7 | 73.3 | 53.3 |
| $B_5A_5$ | 86.0 | 90.7 | 93.0 | 86.0 | 84.8 | 93.9 | 81.8 | 66.7 |
| $B_5C_5$ | 63.0 | 67.4 | 69.6 | 63.0 | 68.6 | 65.7 | 62.9 | 60.0 |
| $B_5D_5$ | 84.4 | 82.2 | 75.6 | 71.1 | 91.2 | 79.4 | 67.6 | 55.9 |
| $C_1A_1$ | 93.2 | 93.2 | 81.8 | 75.0 | 93.9 | 87.9 | 75.8 | 69.7 |
| $C_1B_1$ | 82.6 | 87.0 | 93.5 | 91.3 | 94.3 | 94.3 | 91.4 | 91.4 |
| $C_1D_1$ | 88.4 | 76.7 | 72.1 | 67.4 | 87.5 | 75.0 | 68.8 | 65.6 |
| $C_2A_2$ | 84.4 | 86.7 | 75.6 | 71.1 | 88.2 | 79.4 | 73.5 | 58.8 |
| $C_2B_2$ | 85.2 | 92.6 | 94.4 | 92.6 | 85.0 | 95.0 | 95.0 | 92.5 |
| $C_2D_2$ | 80.0 | 75.6 | 73.3 | 68.9 | 79.4 | 76.5 | 70.6 | 64.7 |
| $C_3A_3$ | 77.8 | 73.3 | 60.0 | 55.6 | 76.5 | 61.8 | 50.0 | 41.2 |
| $C_3B_3$ | 77.6 | 83.7 | 79.6 | 77.6 | 75.7 | 78.4 | 75.7 | 78.4 |
| $C_3D_3$ | 71.7 | 71.7 | 58.7 | 56.5 | 68.6 | 62.9 | 54.3 | 51.4 |
| $C_4A_4$ | 87.8 | 87.8 | 75.6 | 68.3 | 93.5 | 83.9 | 71.0 | 58.1 |
| $C_4B_4$ | 82.9 | 92.7 | 92.7 | 90.2 | 96.8 | 96.8 | 93.5 | 93.5 |
| $C_4D_4$ | 79.5 | 76.9 | 69.2 | 69.2 | 86.2 | 75.9 | 72.4 | 75.9 |
| $C_5A_5$ | 73.0 | 67.6 | 59.5 | 43.2 | 67.9 | 60.7 | 53.6 | 50.0 |
| $C_5B_5$ | 73.9 | 76.1 | 80.4 | 80.4 | 82.9 | 82.9 | 85.7 | 82.9 |
| $C_5D_5$ | 69.2 | 59.0 | 53.8 | 43.6 | 65.5 | 58.6 | 51.7 | 48.3 |
| $D_1A_1$ | 100.0 | 100.0 | 95.7 | 80.9 | 97.1 | 97.1 | 80.0 | 65.7 |
| $D_1B_1$ | 89.8 | 93.9 | 100.0 | 89.8 | 94.6 | 97.3 | 89.2 | 81.1 |
| $D_1C_1$ | 90.7 | 86.0 | 74.4 | 69.8 | 87.5 | 78.1 | 68.8 | 62.5 |
| $D_2A_2$ | 95.2 | 92.9 | 78.6 | 73.8 | 96.9 | 87.5 | 75.0 | 56.3 |
| $D_2B_2$ | 96.1 | 96.1 | 98.0 | 92.2 | 89.5 | 97.4 | 92.1 | 86.8 |
| $D_2C_2$ | 82.2 | 80.0 | 71.1 | 62.2 | 82.4 | 76.5 | 61.8 | 61.8 |
| $D_3A_3$ | 100.0 | 100.0 | 82.9 | 75.6 | 96.8 | 87.1 | 74.2 | 58.1 |
| $D_3B_3$ | 97.8 | 100.0 | 100.0 | 91.1 | 94.1 | 97.1 | 91.2 | 88.2 |
| $D_3C_3$ | 76.1 | 76.1 | 69.6 | 69.6 | 71.4 | 65.7 | 65.7 | 57.1 |
| $D_4A_4$ | 100.0 | 100.0 | 85.0 | 80.0 | 93.3 | 90.0 | 76.7 | 56.7 |
| $D_4B_4$ | 95.0 | 100.0 | 95.0 | 92.5 | 96.7 | 96.7 | 90.0 | 83.3 |
| $D_4C_4$ | 87.2 | 82.1 | 71.8 | 69.2 | 86.2 | 72.4 | 65.5 | 62.1 |
| $D_5A_5$ | 97.2 | 100.0 | 83.3 | 72.2 | 88.9 | 85.2 | 70.4 | 59.3 |
| $D_5B_5$ | 91.1 | 95.6 | 95.6 | 93.3 | 100.0 | 100.0 | 94.1 | 85.3 |
| $D_5C_5$ | 74.4 | 69.2 | 69.2 | 64.1 | 72.4 | 65.5 | 55.2 | 65.5 |

**Table B.4.:** List of evaluation results for two concatenated whole letter trajectories using the UKR-only system for $N^{th} = 3, 4$. This table constitutes the extension of Table 6.2 in Section 6.2.3.

### B.1.3. Two concatenated middle parts of letter trajectories

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1^m B_1^m$ | 92.6 | 88.9 | 85.2 | 70.4 | 95.2 | 85.7 | 71.4 | 61.9 |
| $A_1^m C_1^m$ | 79.2 | 62.5 | 50.0 | 45.8 | 83.3 | 61.1 | 50.0 | n/a |
| $A_1^m D_1^m$ | 88.0 | 72.0 | 68.0 | 64.0 | 84.2 | 73.7 | 68.4 | n/a |
| $A_2^m B_2^m$ | 96.2 | 92.3 | 88.5 | 73.1 | 90.5 | 90.5 | 76.2 | 66.7 |
| $A_2^m C_2^m$ | 75.0 | 75.0 | 62.5 | 54.2 | 68.4 | 63.2 | 52.6 | n/a |
| $A_2^m D_2^m$ | 90.9 | 77.3 | 68.2 | 63.6 | 83.3 | 66.7 | 61.1 | n/a |
| $A_3^m B_3^m$ | 79.2 | 95.8 | 87.5 | 75.0 | 94.4 | 88.9 | 88.9 | n/a |
| $A_3^m C_3^m$ | 54.2 | 58.3 | 54.2 | 50.0 | 55.6 | 50.0 | 44.4 | n/a |
| $A_3^m D_3^m$ | 86.4 | 68.2 | 63.6 | 63.6 | 76.5 | 64.7 | 58.8 | n/a |
| $A_4^m B_4^m$ | 95.5 | 86.4 | 68.2 | 63.6 | 100.0 | 100.0 | 83.3 | n/a |
| $A_4^m C_4^m$ | 81.8 | 72.7 | 54.5 | 68.2 | 88.2 | 64.7 | 52.9 | n/a |
| $A_4^m D_4^m$ | 90.5 | 81.0 | 71.4 | 71.4 | 88.2 | 76.5 | 70.6 | n/a |
| $A_5^m B_5^m$ | 91.3 | 95.7 | 78.3 | 60.9 | 77.8 | 72.2 | 66.7 | n/a |
| $A_5^m C_5^m$ | 60.0 | 65.0 | 60.0 | 60.0 | 46.7 | 46.7 | 40.0 | n/a |
| $A_5^m D_5^m$ | 89.5 | 78.9 | 63.2 | n/a | 66.7 | 60.0 | 53.3 | n/a |

**Table B.5.:** List of evaluation results for two concatenated middle parts of letter trajectories using the UKR-only system for $N^{th} = 3, 4$. This table constitutes the extension of Table 6.3 in Section 6.2.4.

| | $N^{th} = 1$ | | | | $N^{th} = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $B_1^m A_1^m$ | 65.4 | 82.1 | 89.7 | 93.6 | 87.5 | 80.0 | 80.0 | 75.0 |
| $B_1^m C_1^m$ | 73.6 | 84.7 | 87.5 | 83.3 | 94.6 | 83.8 | 70.3 | 59.5 |
| $B_1^m D_1^m$ | 73.7 | 80.3 | 81.6 | 75.0 | 89.7 | 76.9 | 71.8 | 71.8 |
| $B_2^m A_2^m$ | 75.6 | 82.1 | 98.7 | 98.7 | 87.5 | 97.5 | 92.5 | 82.5 |
| $B_2^m C_2^m$ | 61.0 | 68.3 | 74.4 | 72.0 | 76.2 | 71.4 | 71.4 | 64.3 |
| $B_2^m D_2^m$ | 78.2 | 79.5 | 80.8 | 75.6 | 87.5 | 77.5 | 70.0 | 67.5 |
| $B_3^m A_3^m$ | 82.4 | 88.2 | 98.5 | 94.1 | 88.6 | 88.6 | 88.6 | 77.1 |
| $B_3^m C_3^m$ | 52.6 | 63.2 | 60.5 | 63.2 | 64.1 | 64.1 | 66.7 | 64.1 |
| $B_3^m D_3^m$ | 82.9 | 81.4 | 78.6 | 72.9 | 83.3 | 75.0 | 69.4 | 69.4 |
| $B_4^m A_4^m$ | 69.7 | 90.9 | 97.0 | 90.9 | 85.3 | 91.2 | 82.4 | 70.6 |
| $B_4^m C_4^m$ | 65.6 | 73.4 | 75.0 | 71.9 | 69.7 | 78.8 | 66.7 | 51.5 |
| $B_4^m D_4^m$ | 61.3 | 75.8 | 75.8 | 71.0 | 71.9 | 71.9 | 68.8 | 56.3 |
| $B_5^m A_5^m$ | 75.0 | 82.4 | 97.1 | 88.2 | 74.3 | 91.4 | 82.9 | 74.3 |
| $B_5^m C_5^m$ | 54.2 | 63.9 | 68.1 | 69.4 | 54.1 | 70.3 | 67.6 | 62.2 |
| $B_5^m D_5^m$ | 74.3 | 80.0 | 80.0 | 72.9 | 75.0 | 75.0 | 69.4 | 72.2 |
| $C_1^m A_1^m$ | 76.5 | 85.3 | 79.4 | 73.5 | 85.7 | 74.3 | 68.6 | 54.3 |
| $C_1^m B_1^m$ | 73.6 | 91.7 | 97.2 | 90.3 | 100.0 | 94.6 | 91.9 | 91.9 |
| $C_1^m D_1^m$ | 86.4 | 86.4 | 78.8 | 72.7 | 88.2 | 73.5 | 70.6 | 70.6 |
| $C_2^m A_2^m$ | 61.4 | 65.7 | 64.3 | 61.4 | 63.9 | 58.3 | 61.1 | 47.2 |
| $C_2^m B_2^m$ | 65.9 | 74.4 | 82.9 | 78.0 | 81.0 | 78.6 | 81.0 | 85.7 |
| $C_2^m D_2^m$ | 70.0 | 80.0 | 81.4 | 84.3 | 77.8 | 80.6 | 86.1 | 75.0 |
| $C_3^m A_3^m$ | 58.6 | 57.1 | 44.3 | 41.4 | 58.3 | 44.4 | 36.1 | 27.8 |
| $C_3^m B_3^m$ | 59.2 | 69.7 | 67.1 | 63.2 | 71.8 | 66.7 | 66.7 | 69.2 |
| $C_3^m D_3^m$ | 66.7 | 72.2 | 62.5 | 62.5 | 73.0 | 70.3 | 62.2 | 48.6 |
| $C_4^m A_4^m$ | 73.4 | 73.4 | 73.4 | 73.4 | 72.7 | 75.8 | 69.7 | 54.5 |
| $C_4^m B_4^m$ | 70.3 | 85.9 | 85.9 | 87.5 | 81.8 | 90.9 | 87.9 | 90.9 |
| $C_4^m D_4^m$ | 73.3 | 78.3 | 91.7 | 93.3 | 74.2 | 96.8 | 83.9 | 74.2 |
| $C_5^m A_5^m$ | 51.7 | 46.6 | 36.2 | 34.5 | 43.3 | 36.7 | 23.3 | 16.7 |
| $C_5^m B_5^m$ | 54.2 | 56.9 | 56.9 | 55.6 | 51.4 | 56.8 | 62.2 | 59.5 |
| $C_5^m D_5^m$ | 58.3 | 65.0 | 61.7 | 63.3 | 61.3 | 64.5 | 41.9 | 35.5 |
| $D_1^m A_1^m$ | 83.3 | 86.1 | 84.7 | 75.0 | 86.5 | 75.7 | 70.3 | 59.5 |
| $D_1^m B_1^m$ | 77.6 | 80.3 | 82.9 | 78.9 | 89.7 | 79.5 | 71.8 | 64.1 |
| $D_1^m C_1^m$ | 86.4 | 86.4 | 80.3 | 71.2 | 85.3 | 73.5 | 70.6 | 70.6 |
| $D_2^m A_2^m$ | 81.8 | 86.4 | 81.8 | 77.3 | 85.3 | 76.5 | 73.5 | 55.9 |
| $D_2^m B_2^m$ | 80.8 | 84.6 | 83.3 | 84.6 | 95.0 | 85.0 | 85.0 | 77.5 |
| $D_2^m C_2^m$ | 64.3 | 72.9 | 71.4 | 74.3 | 72.2 | 72.2 | 77.8 | 66.7 |
| $D_3^m A_3^m$ | 92.2 | 85.9 | 81.3 | 71.9 | 84.8 | 72.7 | 69.7 | 51.5 |
| $D_3^m B_3^m$ | 90.0 | 92.9 | 91.4 | 82.9 | 94.4 | 83.3 | 86.1 | 72.2 |
| $D_3^m C_3^m$ | 59.7 | 65.3 | 58.3 | 61.1 | 64.9 | 62.2 | 64.9 | 62.2 |
| $D_4^m A_4^m$ | 72.6 | 80.6 | 80.6 | 74.2 | 75.0 | 71.9 | 65.6 | 50.0 |
| $D_4^m B_4^m$ | 69.4 | 88.7 | 83.9 | 83.9 | 81.3 | 81.3 | 78.1 | 71.9 |
| $D_4^m C_4^m$ | 65.0 | 68.3 | 71.7 | 81.7 | 61.3 | 77.4 | 67.7 | 64.5 |
| $D_5^m A_5^m$ | 83.9 | 85.7 | 75.0 | 71.4 | 86.2 | 72.4 | 65.5 | 51.7 |
| $D_5^m B_5^m$ | 77.1 | 81.4 | 84.3 | 84.3 | 75.0 | 83.3 | 75.0 | 66.7 |
| $D_5^m C_5^m$ | 58.3 | 65.0 | 61.7 | 63.3 | 61.3 | 64.5 | 67.7 | 71.0 |

**Table B.6.:** List of evaluation results for two concatenated middle parts of letter trajectories using the UKR-only system for $N^{th} = 1, 2$. This table constitutes the extension of Table 6.3 in Section 6.2.4.

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $B_1^m A_1^m$ | 88.9 | 88.9 | 74.1 | 59.3 | 76.2 | 76.2 | 57.1 | 52.4 |
| $B_1^m C_1^m$ | 88.0 | 72.0 | 60.0 | 56.0 | 84.2 | 57.9 | 57.9 | n/a |
| $B_1^m D_1^m$ | 80.8 | 73.1 | 73.1 | 73.1 | 75.0 | 75.0 | 75.0 | 75.0 |
| $B_2^m A_2^m$ | 92.3 | 84.6 | 73.1 | 57.7 | 95.2 | 85.7 | 57.1 | 57.1 |
| $B_2^m C_2^m$ | 75.0 | 71.4 | 53.6 | 53.6 | 72.7 | 68.2 | 54.5 | 54.5 |
| $B_2^m D_2^m$ | 84.6 | 69.2 | 69.2 | 69.2 | 76.2 | 71.4 | 66.7 | 71.4 |
| $B_3^m A_3^m$ | 75.0 | 87.5 | 79.2 | 62.5 | 94.4 | 88.9 | 61.1 | n/a |
| $B_3^m C_3^m$ | 50.0 | 65.4 | 61.5 | 53.8 | 65.0 | 65.0 | 60.0 | 55.0 |
| $B_3^m D_3^m$ | 66.7 | 70.8 | 70.8 | 70.8 | 73.7 | 73.7 | 73.7 | n/a |
| $B_4^m A_4^m$ | 90.9 | 72.7 | 63.6 | 50.0 | 77.8 | 72.2 | 55.6 | n/a |
| $B_4^m C_4^m$ | 72.7 | 63.6 | 50.0 | 50.0 | 82.4 | 64.7 | 58.8 | n/a |
| $B_4^m D_4^m$ | 81.0 | 71.4 | 66.7 | 66.7 | 70.6 | 70.6 | 70.6 | n/a |
| $B_5^m A_5^m$ | 91.3 | 87.0 | 69.6 | 65.2 | 83.3 | 77.8 | 61.1 | n/a |
| $B_5^m C_5^m$ | 64.0 | 68.0 | 64.0 | 60.0 | 68.4 | 57.9 | 57.9 | n/a |
| $B_5^m D_5^m$ | 75.0 | 70.8 | 70.8 | 70.8 | 78.9 | 73.7 | 68.4 | n/a |
| $C_1^m A_1^m$ | 79.2 | 70.8 | 58.3 | 50.0 | 72.2 | 61.1 | 55.6 | n/a |
| $C_1^m B_1^m$ | 100.0 | 92.0 | 92.0 | 92.0 | 94.7 | 89.5 | 89.5 | n/a |
| $C_1^m D_1^m$ | 82.6 | 69.6 | 69.6 | 69.6 | 76.5 | 70.6 | 70.6 | n/a |
| $C_2^m A_2^m$ | 58.3 | 66.7 | 50.0 | 50.0 | 63.2 | 57.9 | 47.4 | n/a |
| $C_2^m B_2^m$ | 82.1 | 85.7 | 85.7 | 85.7 | 81.8 | 86.4 | 90.9 | 90.9 |
| $C_2^m D_2^m$ | 79.2 | 91.7 | 70.8 | 62.5 | 84.2 | 84.2 | 68.4 | n/a |
| $C_3^m A_3^m$ | 45.8 | 37.5 | 25.0 | 25.0 | 44.4 | 27.8 | 27.8 | n/a |
| $C_3^m B_3^m$ | 57.7 | 69.2 | 73.1 | 69.2 | 70.0 | 70.0 | 70.0 | 70.0 |
| $C_3^m D_3^m$ | 66.7 | 58.3 | 45.8 | 41.7 | 68.4 | 52.6 | 47.4 | n/a |
| $C_4^m A_4^m$ | 68.2 | 72.7 | 54.5 | 54.5 | 70.6 | 58.8 | 52.9 | n/a |
| $C_4^m B_4^m$ | 90.9 | 90.9 | 90.9 | 90.9 | 88.2 | 88.2 | 88.2 | n/a |
| $C_4^m D_4^m$ | 85.7 | 81.0 | 76.2 | 76.2 | 100.0 | 75.0 | 75.0 | n/a |
| $C_5^m A_5^m$ | 35.0 | 20.0 | 15.0 | 15.0 | 33.3 | 13.3 | 13.3 | n/a |
| $C_5^m B_5^m$ | 52.0 | 56.0 | 56.0 | 68.0 | 57.9 | 68.4 | 68.4 | n/a |
| $C_5^m D_5^m$ | 61.9 | 42.9 | 28.6 | 28.6 | 56.3 | 31.3 | 31.3 | n/a |
| $D_1^m A_1^m$ | 80.0 | 68.0 | 52.0 | 44.0 | 73.7 | 68.4 | 47.4 | n/a |
| $D_1^m B_1^m$ | 80.8 | 73.1 | 61.5 | 61.5 | 80.0 | 65.0 | 65.0 | 65.0 |
| $D_1^m C_1^m$ | 78.3 | 65.2 | 65.2 | 65.2 | 70.6 | 64.7 | 70.6 | n/a |
| $D_2^m A_2^m$ | 77.3 | 72.7 | 54.5 | 50.0 | 72.2 | 55.6 | 44.4 | n/a |
| $D_2^m B_2^m$ | 88.5 | 84.6 | 84.6 | 73.1 | 81.0 | 85.7 | 61.9 | 61.9 |
| $D_2^m C_2^m$ | 79.2 | 79.2 | 66.7 | 66.7 | 63.2 | 63.2 | 57.9 | n/a |
| $D_3^m A_3^m$ | 77.3 | 63.6 | 50.0 | 50.0 | 70.6 | 64.7 | 52.9 | n/a |
| $D_3^m B_3^m$ | 79.2 | 83.3 | 66.7 | 54.2 | 94.7 | 84.2 | 68.4 | n/a |
| $D_3^m C_3^m$ | 58.3 | 66.7 | 66.7 | 66.7 | 63.2 | 63.2 | 68.4 | n/a |
| $D_4^m A_4^m$ | 76.2 | 66.7 | 47.6 | 42.9 | 70.6 | 64.7 | 47.1 | n/a |
| $D_4^m B_4^m$ | 81.0 | 76.2 | 76.2 | 66.7 | 82.4 | 76.5 | 70.6 | n/a |
| $D_4^m C_4^m$ | 66.7 | 71.4 | 66.7 | 66.7 | 81.3 | 68.8 | 68.8 | n/a |
| $D_5^m A_5^m$ | 68.4 | 63.2 | 47.4 | n/a | 73.3 | 53.3 | 53.3 | n/a |
| $D_5^m B_5^m$ | 75.0 | 75.0 | 66.7 | 62.5 | 89.5 | 73.7 | 57.9 | n/a |
| $D_5^m C_5^m$ | 57.1 | 61.9 | 66.7 | 66.7 | 62.5 | 68.8 | 75.0 | n/a |

**Table B.7.:** List of evaluation results for two concatenated middle parts of letter trajectories using the UKR-only system for $N^{th} = 3, 4$. This table constitutes the extension of Table 6.3 in Section 6.2.4.

## B.2. Recognition with the combined UKR/CLM system

### B.2.1. Single whole letter trajectories ($N^{th} = 3, 4$)

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_5$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a | 100.0 (0.0) | 69.2 (0.0) | n/a | n/a |
| $B_1$ | 84.6 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 88.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 90.0 (0.0) | 90.0 (0.0) |
| $B_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 93.3 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 96.6 (2.3) | 81.8 (0.0) |
| $B_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 79.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $B_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $B_5$ | 96.2 (0.0) | 96.2 (0.0) | 96.2 (0.0) | 88.5 (0.0) | 95.0 (0.0) | 95.0 (0.0) | 95.0 (0.0) | 90.0 (0.0) |
| $C_1$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $C_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $C_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $C_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $C_5$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_1$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a | 100.0 (0.0) | 100.0 (0.0) | n/a | n/a |
| $D_5$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a | 100.0 (0.0) | 100.0 (0.0) | n/a | n/a |

**Table B.8.:** List of evaluation results for single whole letter trajectories using the combinded UKR/CLM system for $N^{th} = 3, 4$. This table constitutes the extension of Table 6.4 in Section 6.3.5. The values are: mean (stdvar).

### B.2.2. Two concatenated whole letter trajectories

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1B_1$ | 88.0 (0.0) | 98.0 (0.0) | 84.0 (0.0) | 76.0 (0.0) | 94.7 (0.0) | 89.5 (0.0) | 78.9 (0.0) | 71.1 (0.0) |
| $A_1C_1$ | 100.0 (0.0) | 97.7 (0.0) | 79.0 (1.1) | 79.5 (0.0) | 100.0 (0.0) | 93.9 (0.0) | 81.8 (0.0) | 69.7 (0.0) |
| $A_1D_1$ | 97.9 (0.0) | 91.5 (0.0) | 86.2 (1.2) | 79.3 (1.1) | 97.1 (0.0) | 88.6 (0.0) | 82.9 (0.0) | 65.7 (0.0) |
| $A_2B_2$ | 98.0 (0.0) | 98.0 (0.0) | 88.2 (0.0) | 86.3 (0.0) | 90.1 (1.3) | 97.4 (0.0) | 94.7 (0.0) | 61.2 (1.3) |
| $A_2C_2$ | 100.0 (0.0) | 100.0 (0.0) | 86.7 (0.0) | 82.8 (1.1) | 100.0 (0.0) | 97.8 (1.5) | 79.4 (0.0) | 85.3 (0.0) |
| $A_2D_2$ | 97.6 (0.0) | 92.9 (0.0) | 85.7 (0.0) | 81.0 (0.0) | 96.9 (0.0) | 87.5 (0.0) | 81.3 (0.0) | 62.5 (0.0) |
| $A_3B_3$ | 97.7 (0.0) | 90.9 (0.0) | 86.4 (0.0) | 84.1 (0.0) | 96.2 (1.5) | 75.8 (0.0) | 77.3 (3.0) | 54.5 (0.0) |
| $A_3C_3$ | 100.0 (0.0) | 100.0 (0.0) | 84.4 (0.0) | 82.2 (0.0) | 100.0 (0.0) | 85.3 (0.0) | 79.4 (0.0) | 82.4 (0.0) |
| $A_3D_3$ | 97.6 (0.0) | 90.2 (0.0) | 85.4 (0.0) | 78.0 (0.0) | 96.8 (0.0) | 87.1 (0.0) | 80.6 (0.0) | 45.2 (0.0) |
| $A_4B_4$ | 88.1 (4.8) | 88.1 (0.0) | 73.8 (0.0) | 78.6 (0.0) | 84.4 (0.0) | 77.3 (1.6) | 78.1 (0.0) | 76.6 (3.1) |
| $A_4C_4$ | 100.0 (0.0) | 97.0 (1.2) | 82.9 (0.0) | 82.9 (0.0) | 100.0 (0.0) | 83.9 (0.0) | 77.4 (0.0) | 77.4 (0.0) |
| $A_4D_4$ | 97.5 (0.0) | 90.6 (1.3) | 85.0 (0.0) | 77.5 (0.0) | 96.7 (0.0) | 86.7 (0.0) | 76.7 (0.0) | 66.7 (0.0) |
| $A_5B_5$ | 88.4 (0.0) | 93.0 (0.0) | 86.0 (0.0) | 79.1 (0.0) | 93.9 (0.0) | 81.8 (0.0) | 81.8 (0.0) | 51.5 (0.0) |
| $A_5C_5$ | 100.0 (0.0) | 83.8 (0.0) | 81.1 (0.0) | 78.4 (0.0) | 100.0 (0.0) | 82.1 (0.0) | 78.6 (0.0) | 50.0 (0.0) |
| $A_5D_5$ | 97.2 (0.0) | 83.3 (0.0) | 83.3 (0.0) | 75.0 (0.0) | 92.6 (0.0) | 85.2 (0.0) | 74.1 (0.0) | 40.7 (0.0) |

**Table B.9.:** List of evaluation results for two concatenated whole letter trajectories using the combinded UKR/CLM system for $N^{th} = 3, 4$. This table constitutes the extension of Table 6.5 in Section 6.3.6. The values are: mean (stdvar).

| | $N^{th} = 1$ | | | | $N^{th} = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $B_1A_1$ | 82.4 (5.7) | 82.6 (0.3) | 81.3 (0.3) | 77.0 (0.0) | 75.7 (0.0) | 77.0 (0.0) | 74.3 (0.0) | 62.2 (0.0) |
| $B_1C_1$ | 91.9 (0.0) | 95.6 (0.0) | 100.0 (0.0) | 99.3 (0.0) | 88.2 (0.0) | 100.0 (0.0) | 85.3 (0.0) | 79.4 (0.0) |
| $B_1D_1$ | 91.8 (0.0) | 91.8 (2.8) | 95.9 (0.0) | 93.8 (0.0) | 87.7 (0.0) | 94.5 (0.0) | 91.8 (0.0) | 87.7 (0.0) |
| $B_2A_2$ | 77.2 (1.3) | 87.6 (0.3) | 76.2 (0.0) | 76.2 (0.0) | 85.2 (2.0) | 75.0 (0.0) | 73.7 (0.0) | 65.8 (0.0) |
| $B_2C_2$ | 90.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 98.7 (0.0) | 98.7 (0.0) | 81.0 (0.0) | 81.0 (0.0) |
| $B_2D_2$ | 90.1 (0.0) | 98.0 (0.0) | 96.7 (0.0) | 94.7 (0.0) | 97.4 (0.0) | 92.1 (0.0) | 83.6 (0.8) | 82.9 (0.0) |
| $B_3A_3$ | 80.2 (6.1) | 91.6 (0.0) | 84.4 (0.4) | 77.9 (0.0) | 92.4 (0.0) | 77.3 (0.0) | 68.2 (0.0) | 80.3 (3.0) |
| $B_3C_3$ | 94.4 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 83.6 (0.0) | 79.5 (0.0) |
| $B_3D_3$ | 92.7 (0.4) | 97.0 (0.0) | 95.5 (0.0) | 93.3 (0.0) | 98.5 (0.0) | 94.1 (0.0) | 86.8 (0.0) | 86.8 (0.0) |
| $B_4A_4$ | 85.4 (0.0) | 95.1 (0.0) | 93.7 (2.0) | 93.5 (0.0) | 95.2 (2.3) | 93.5 (0.0) | 93.5 (0.0) | 91.9 (0.0) |
| $B_4C_4$ | 100.0 (0.0) | 100.0 (0.0) | 91.9 (5.4) | 84.3 (0.0) | 100.0 (0.0) | 86.5 (2.1) | 83.6 (0.0) | 77.0 (0.0) |
| $B_4D_4$ | 95.5 (6.2) | 93.8 (0.8) | 90.9 (0.5) | 88.8 (0.0) | 96.2 (0.8) | 89.4 (0.8) | 88.1 (0.0) | 88.1 (0.0) |
| $B_5A_5$ | 61.7 (0.0) | 54.3 (0.5) | 57.2 (0.4) | 57.0 (0.0) | 57.4 (1.5) | 39.1 (0.0) | 25.0 (0.0) | 25.0 (0.0) |
| $B_5C_5$ | 79.7 (0.0) | 81.2 (0.0) | 82.1 (0.4) | 81.9 (0.0) | 81.2 (0.0) | 65.2 (0.0) | 43.5 (0.0) | 43.5 (0.0) |
| $B_5D_5$ | 78.9 (0.0) | 79.1 (1.9) | 77.6 (0.4) | 75.2 (0.0) | 79.1 (0.0) | 58.2 (0.0) | 41.8 (0.0) | 41.8 (0.0) |
| $C_1A_1$ | 89.5 (0.5) | 96.9 (0.0) | 96.1 (0.0) | 93.8 (0.0) | 96.9 (0.0) | 93.8 (0.0) | 92.2 (0.0) | 87.5 (0.0) |
| $C_1B_1$ | 91.2 (0.0) | 94.9 (0.0) | 95.6 (0.0) | 97.8 (0.0) | 90.4 (4.2) | 97.1 (0.0) | 98.5 (0.0) | 89.7 (0.0) |
| $C_1D_1$ | 99.0 (0.4) | 97.4 (0.4) | 96.0 (0.0) | 95.2 (0.0) | 98.4 (0.0) | 95.6 (0.8) | 92.1 (0.0) | 92.1 (0.0) |
| $C_2A_2$ | 88.2 (0.7) | 93.6 (0.4) | 95.5 (0.0) | 93.2 (0.0) | 94.0 (0.0) | 94.0 (0.0) | 92.5 (0.0) | 86.6 (0.0) |
| $C_2B_2$ | 90.5 (0.0) | 98.1 (0.0) | 97.5 (0.0) | 97.5 (0.0) | 97.5 (0.0) | 96.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_2D_2$ | 99.2 (0.0) | 97.0 (0.0) | 95.5 (0.0) | 93.4 (0.4) | 97.8 (0.9) | 94.0 (0.0) | 94.0 (0.0) | 94.0 (0.0) |
| $C_3A_3$ | 86.7 (0.4) | 97.0 (0.0) | 95.5 (0.0) | 94.5 (0.4) | 96.6 (0.7) | 95.5 (0.0) | 95.5 (0.0) | 88.1 (0.0) |
| $C_3B_3$ | 93.6 (0.3) | 97.9 (0.0) | 97.9 (0.0) | 95.1 (0.0) | 98.6 (0.0) | 95.9 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_3D_3$ | 99.3 (0.0) | 97.8 (0.0) | 96.3 (0.0) | 95.6 (0.0) | 98.6 (0.0) | 95.7 (0.0) | 94.2 (0.0) | 100.0 (0.0) |
| $C_4A_4$ | 86.9 (5.5) | 91.8 (0.0) | 95.1 (0.0) | 94.3 (0.0) | 95.1 (0.0) | 95.1 (0.0) | 95.1 (0.0) | 86.9 (0.0) |
| $C_4B_4$ | 93.4 (6.7) | 91.7 (0.0) | 92.6 (0.0) | 95.0 (0.0) | 91.8 (0.0) | 95.1 (0.0) | 100.0 (0.0) | 98.4 (0.0) |
| $C_4D_4$ | 99.1 (0.0) | 97.8 (0.5) | 96.3 (0.4) | 95.7 (0.0) | 98.3 (0.0) | 98.3 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_5A_5$ | 83.6 (0.0) | 82.7 (0.0) | 94.5 (0.0) | 93.6 (0.0) | 83.6 (0.0) | 94.5 (0.0) | 99.5 (0.9) | 80.0 (7.3) |
| $C_5B_5$ | 81.0 (2.5) | 81.2 (0.0) | 82.6 (0.0) | 81.9 (0.0) | 81.2 (0.0) | 65.2 (0.0) | 66.7 (0.0) | 60.9 (0.0) |
| $C_5D_5$ | 99.1 (0.0) | 97.6 (0.4) | 96.5 (0.0) | 95.7 (0.0) | 98.3 (0.0) | 94.8 (0.0) | 98.7 (2.6) | 100.0 (0.0) |
| $D_1A_1$ | 94.4 (4.8) | 97.8 (0.0) | 96.4 (0.0) | 95.7 (0.0) | 98.6 (0.0) | 95.7 (0.0) | 94.2 (0.0) | 92.8 (0.0) |
| $D_1B_1$ | 84.9 (0.0) | 90.6 (0.3) | 93.2 (0.0) | 95.2 (0.0) | 89.0 (0.0) | 94.5 (0.0) | 97.3 (0.0) | 98.6 (0.0) |
| $D_1C_1$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 89.3 (0.8) | 81.0 (0.0) |
| $D_2A_2$ | 90.7 (5.2) | 96.8 (0.0) | 95.2 (0.0) | 93.8 (0.4) | 98.4 (0.0) | 94.1 (0.8) | 93.8 (0.0) | 89.1 (0.0) |
| $D_2B_2$ | 88.7 (0.0) | 88.7 (0.0) | 90.1 (0.0) | 91.4 (0.0) | 84.2 (0.0) | 89.5 (0.0) | 93.4 (0.0) | 98.7 (0.0) |
| $D_2C_2$ | 100.0 (0.0) | 100.0 (0.0) | 97.2 (2.2) | 96.1 (0.7) | 100.0 (0.0) | 97.4 (1.9) | 89.2 (0.7) | 91.4 (0.7) |
| $D_3A_3$ | 92.1 (7.3) | 96.7 (0.0) | 95.9 (0.0) | 94.3 (0.0) | 98.4 (0.0) | 94.4 (0.9) | 95.2 (0.0) | 90.3 (0.0) |
| $D_3B_3$ | 92.4 (0.7) | 86.6 (0.0) | 88.1 (0.0) | 90.3 (0.0) | 86.8 (0.0) | 89.7 (0.0) | 91.2 (0.0) | 98.5 (0.0) |
| $D_3C_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 97.8 (1.2) | 100.0 (0.0) | 98.2 (1.8) | 96.7 (1.4) | 82.2 (3.2) |
| $D_4A_4$ | 87.8 (6.3) | 94.9 (0.0) | 94.9 (0.0) | 94.9 (0.0) | 96.6 (0.0) | 94.9 (0.0) | 93.2 (0.0) | 88.1 (0.0) |
| $D_4B_4$ | 87.9 (0.0) | 90.5 (0.0) | 92.2 (0.0) | 94.8 (0.0) | 89.8 (0.0) | 94.9 (0.0) | 98.3 (0.0) | 96.6 (0.0) |
| $D_4C_4$ | 100.0 (0.0) | 100.0 (0.0) | 97.4 (3.0) | 98.7 (2.6) | 100.0 (0.0) | 97.8 (4.3) | 90.9 (7.1) | 74.1 (0.0) |
| $D_5A_5$ | 82.9 (0.0) | 81.0 (0.0) | 95.2 (0.0) | 95.2 (0.0) | 81.1 (0.0) | 94.3 (0.0) | 88.2 (8.5) | 86.8 (0.0) |
| $D_5B_5$ | 78.9 (0.0) | 80.5 (0.0) | 82.9 (0.4) | 82.9 (0.9) | 80.6 (0.0) | 64.2 (0.0) | 65.7 (0.0) | 59.7 (0.0) |
| $D_5C_5$ | 100.0 (0.0) | 97.0 (3.5) | 94.8 (0.0) | 96.7 (0.4) | 93.1 (0.0) | 96.6 (0.0) | 100.0 (0.0) | 86.2 (5.1) |

**Table B.10.:** List of evaluation results for two concatenated whole letter trajectories using the combinded UKR/CLM system for $N^{th} = 1, 2$. This table constitutes the extension of Table 6.5 in Section 6.3.6. The values are: mean (stdvar).

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $B_1A_1$ | 70.0 (0.0) | 76.0 (0.0) | 48.0 (0.0) | 86.0 (0.0) | 76.3 (0.0) | 57.9 (7.1) | 84.2 (0.0) | 73.7 (0.0) |
| $B_1C_1$ | 89.1 (0.0) | 88.6 (2.1) | 80.4 (0.0) | 69.6 (0.0) | 100.0 (0.0) | 90.0 (6.8) | 77.1 (5.7) | 56.4 (9.1) |
| $B_1D_1$ | 87.8 (0.0) | 91.8 (0.0) | 89.8 (0.0) | 81.6 (0.0) | 97.3 (0.0) | 89.2 (0.0) | 83.8 (0.0) | 67.6 (0.0) |
| $B_2A_2$ | 76.5 (0.0) | 72.5 (0.0) | 58.8 (0.0) | 64.7 (0.0) | 76.3 (0.0) | 65.8 (0.0) | 71.1 (0.0) | 65.8 (0.0) |
| $B_2C_2$ | 100.0 (0.0) | 93.5 (2.4) | 82.4 (1.1) | 85.2 (0.0) | 100.0 (0.0) | 82.5 (0.0) | 76.3 (2.5) | 85.0 (0.0) |
| $B_2D_2$ | 98.0 (0.0) | 92.2 (0.0) | 86.3 (0.0) | 84.3 (0.0) | 94.7 (0.0) | 86.8 (0.0) | 84.2 (0.0) | 71.1 (0.0) |
| $B_3A_3$ | 88.6 (0.0) | 79.0 (5.7) | 86.4 (0.0) | 79.5 (0.0) | 78.8 (0.0) | 87.9 (0.0) | 78.8 (0.0) | 45.5 (0.0) |
| $B_3C_3$ | 100.0 (0.0) | 85.2 (1.0) | 79.6 (0.0) | 81.6 (0.0) | 100.0 (0.0) | 83.8 (0.0) | 81.1 (0.0) | 83.8 (0.0) |
| $B_3D_3$ | 94.4 (6.7) | 91.1 (0.0) | 86.7 (0.0) | 80.0 (0.0) | 85.3 (0.0) | 88.2 (0.0) | 76.5 (0.0) | 73.5 (0.0) |
| $B_4A_4$ | 81.0 (0.0) | 90.5 (0.0) | 88.1 (0.0) | 81.0 (0.0) | 96.9 (0.0) | 90.6 (0.0) | 84.4 (0.0) | 43.8 (0.0) |
| $B_4C_4$ | 93.3 (8.3) | 82.9 (0.0) | 78.0 (0.0) | 82.9 (0.0) | 89.5 (7.2) | 78.2 (1.6) | 83.9 (0.0) | 77.4 (0.0) |
| $B_4D_4$ | 90.0 (0.0) | 87.5 (0.0) | 87.5 (0.0) | 77.5 (0.0) | 93.3 (2.7) | 86.7 (0.0) | 76.7 (0.0) | 36.7 (0.0) |
| $B_5A_5$ | 64.5 (1.2) | 74.4 (0.0) | 41.9 (0.0) | 74.4 (0.0) | 76.5 (1.5) | 66.7 (0.0) | 68.2 (1.7) | 71.2 (3.0) |
| $B_5C_5$ | 88.0 (1.3) | 95.1 (5.4) | 81.0 (1.1) | 79.3 (5.2) | 97.1 (0.0) | 71.4 (0.0) | 77.1 (0.0) | 78.6 (4.9) |
| $B_5D_5$ | 85.0 (1.1) | 88.9 (0.0) | 86.7 (0.0) | 77.8 (0.0) | 94.1 (0.0) | 88.2 (0.0) | 88.2 (0.0) | 73.5 (0.0) |
| $C_1A_1$ | 95.5 (0.0) | 90.9 (0.0) | 86.4 (0.0) | 81.8 (0.0) | 96.2 (1.5) | 87.9 (0.0) | 81.8 (0.0) | 57.6 (0.0) |
| $C_1B_1$ | 87.0 (0.0) | 91.3 (0.0) | 87.0 (0.0) | 76.1 (0.0) | 97.1 (0.0) | 88.6 (0.0) | 77.1 (0.0) | 51.4 (0.0) |
| $C_1D_1$ | 98.3 (1.2) | 90.7 (0.0) | 90.7 (0.0) | 90.7 (0.0) | 98.4 (1.8) | 90.6 (0.0) | 90.6 (0.0) | 56.3 (0.0) |
| $C_2A_2$ | 94.4 (1.3) | 93.3 (0.0) | 86.7 (0.0) | 80.0 (0.0) | 97.1 (0.0) | 88.2 (0.0) | 82.4 (0.0) | 47.1 (0.0) |
| $C_2B_2$ | 98.1 (0.0) | 100.0 (0.0) | 98.1 (0.0) | 88.0 (1.1) | 97.5 (0.0) | 100.0 (0.0) | 92.5 (0.0) | 77.5 (0.0) |
| $C_2D_2$ | 96.1 (1.1) | 93.3 (0.0) | 93.3 (0.0) | 93.9 (1.1) | 95.6 (1.7) | 94.1 (0.0) | 94.1 (0.0) | 61.0 (13.0) |
| $C_3A_3$ | 97.8 (0.0) | 95.6 (0.0) | 86.7 (0.0) | 81.1 (1.3) | 97.1 (0.0) | 91.2 (0.0) | 82.4 (0.0) | 52.9 (0.0) |
| $C_3B_3$ | 98.0 (0.0) | 100.0 (0.0) | 98.0 (0.0) | 83.7 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 91.9 (0.0) | 75.7 (0.0) |
| $C_3D_3$ | 97.8 (0.0) | 96.2 (1.1) | 97.8 (0.0) | 95.7 (0.0) | 97.1 (0.0) | 100.0 (0.0) | 97.1 (0.0) | 67.1 (4.9) |
| $C_4A_4$ | 97.6 (0.0) | 97.6 (0.0) | 85.4 (0.0) | 79.9 (1.2) | 96.8 (0.0) | 87.1 (0.0) | 80.6 (0.0) | 48.4 (0.0) |
| $C_4B_4$ | 92.7 (0.0) | 100.0 (0.0) | 97.6 (0.0) | 90.2 (0.0) | 97.6 (3.1) | 100.0 (0.0) | 93.5 (0.0) | 51.6 (0.0) |
| $C_4D_4$ | 97.4 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 80.2 (26.2) |
| $C_5A_5$ | 97.3 (0.0) | 91.9 (0.0) | 83.8 (0.0) | 56.8 (0.0) | 96.4 (0.0) | 72.3 (1.8) | 60.7 (0.0) | 50.0 (0.0) |
| $C_5B_5$ | 89.1 (0.0) | 97.8 (0.0) | 91.3 (0.0) | 80.4 (0.0) | 97.1 (0.0) | 91.4 (0.0) | 85.7 (0.0) | 48.6 (0.0) |
| $C_5D_5$ | 98.1 (1.3) | 98.7 (2.6) | 98.1 (1.3) | 88.5 (2.6) | 96.6 (0.0) | 96.6 (0.0) | 77.6 (2.0) | 48.3 (0.0) |
| $D_1A_1$ | 97.9 (0.0) | 94.1 (1.1) | 91.5 (0.0) | 87.2 (0.0) | 97.1 (0.0) | 91.4 (0.0) | 88.6 (0.0) | 78.6 (1.6) |
| $D_1B_1$ | 91.8 (0.0) | 98.0 (0.0) | 98.0 (0.0) | 85.7 (0.0) | 91.9 (0.0) | 100.0 (0.0) | 89.2 (0.0) | 81.1 (0.0) |
| $D_1C_1$ | 100.0 (0.0) | 90.1 (2.2) | 83.7 (0.0) | 90.7 (0.0) | 99.2 (1.6) | 77.3 (3.0) | 90.6 (0.0) | 93.8 (0.0) |
| $D_2A_2$ | 97.6 (0.0) | 95.2 (0.0) | 88.1 (0.0) | 85.7 (0.0) | 96.9 (0.0) | 90.6 (0.0) | 84.4 (0.0) | 53.1 (0.0) |
| $D_2B_2$ | 88.2 (0.0) | 94.1 (0.0) | 100.0 (0.0) | 92.2 (0.0) | 89.5 (0.0) | 100.0 (0.0) | 97.4 (0.0) | 81.6 (0.0) |
| $D_2C_2$ | 95.0 (3.3) | 88.9 (1.8) | 93.3 (0.0) | 93.3 (0.0) | 97.1 (3.4) | 92.6 (1.7) | 100.0 (0.0) | 94.1 (0.0) |
| $D_3A_3$ | 97.6 (0.0) | 95.1 (0.0) | 87.8 (0.0) | 85.4 (0.0) | 96.8 (0.0) | 90.3 (0.0) | 87.1 (0.0) | 48.4 (0.0) |
| $D_3B_3$ | 86.7 (0.0) | 93.3 (0.0) | 97.8 (0.0) | 86.7 (0.0) | 88.2 (0.0) | 97.1 (0.0) | 97.1 (0.0) | 88.2 (0.0) |
| $D_3C_3$ | 97.3 (2.7) | 94.0 (4.5) | 78.3 (0.0) | 83.7 (1.3) | 94.3 (0.0) | 87.1 (6.8) | 80.0 (0.0) | 88.6 (0.0) |
| $D_4A_4$ | 95.6 (1.3) | 92.5 (0.0) | 87.5 (0.0) | 85.0 (0.0) | 96.7 (0.0) | 90.0 (0.0) | 86.7 (0.0) | 46.7 (0.0) |
| $D_4B_4$ | 90.0 (0.0) | 97.5 (0.0) | 97.5 (0.0) | 90.0 (0.0) | 90.8 (1.7) | 100.0 (0.0) | 93.3 (0.0) | 50.0 (0.0) |
| $D_4C_4$ | 96.8 (3.8) | 85.3 (3.2) | 78.8 (9.0) | 86.5 (2.5) | 92.2 (5.9) | 75.9 (0.0) | 82.8 (0.0) | 51.7 (0.0) |
| $D_5A_5$ | 97.2 (0.0) | 91.7 (0.0) | 86.1 (0.0) | 79.2 (4.8) | 96.3 (0.0) | 88.9 (0.0) | 63.0 (0.0) | 46.3 (4.8) |
| $D_5B_5$ | 88.9 (0.0) | 97.8 (0.0) | 93.3 (0.0) | 84.4 (0.0) | 91.2 (0.0) | 94.1 (0.0) | 88.2 (0.0) | 79.4 (0.0) |
| $D_5C_5$ | 92.3 (0.0) | 98.1 (3.8) | 91.7 (2.5) | 89.7 (0.0) | 94.0 (1.7) | 95.7 (1.7) | 93.1 (0.0) | 70.7 (21.9) |

**Table B.11.:** List of evaluation results for two concatenated whole letter trajectories using the combinded UKR/CLM system for $N^{th} = 3, 4$. This table constitutes the extension of Table 6.5 in Section 6.3.6. The values are: mean (stdvar).

### B.2.3. Two concatenated middle parts of letter trajectories

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1^m B_1^m$ | 100.0 (0.0) | 91.7 (1.9) | 81.5 (0.0) | 70.4 (0.0) | 90.5 (0.0) | 95.2 (0.0) | 66.7 (0.0) | 52.4 (0.0) |
| $A_1^m C_1^m$ | 87.5 (0.0) | 75.0 (0.0) | 66.7 (0.0) | 45.8 (0.0) | 83.3 (0.0) | 72.2 (0.0) | 44.4 (0.0) | n/a |
| $A_1^m D_1^m$ | 84.0 (0.0) | 76.0 (0.0) | 64.0 (0.0) | 48.0 (0.0) | 78.9 (0.0) | 63.2 (0.0) | 47.4 (0.0) | n/a |
| $A_2^m B_2^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 88.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 85.7 (0.0) | 57.1 (0.0) |
| $A_2^m C_2^m$ | 91.7 (0.0) | 80.2 (2.1) | 94.8 (2.1) | 79.2 (0.0) | 84.2 (0.0) | 73.7 (0.0) | 52.6 (0.0) | n/a |
| $A_2^m D_2^m$ | 86.4 (0.0) | 72.7 (0.0) | 63.6 (0.0) | 50.0 (0.0) | 83.3 (0.0) | 66.7 (0.0) | 50.0 (0.0) | n/a |
| $A_3^m B_3^m$ | 95.8 (0.0) | 95.8 (0.0) | 93.8 (7.2) | 54.2 (0.0) | 38.9 (0.0) | 33.3 (0.0) | 0.0 (0.0) | n/a |
| $A_3^m C_3^m$ | 79.2 (0.0) | 79.2 (0.0) | 75.0 (0.0) | 58.3 (0.0) | 77.8 (0.0) | 83.3 (0.0) | 61.1 (0.0) | n/a |
| $A_3^m D_3^m$ | 86.4 (0.0) | 72.7 (0.0) | 50.0 (0.0) | 50.0 (0.0) | 100.0 (0.0) | 70.6 (0.0) | 52.9 (0.0) | n/a |
| $A_4^m B_4^m$ | 95.5 (0.0) | 95.5 (0.0) | 77.3 (0.0) | 54.5 (9.1) | 98.6 (2.8) | 87.5 (8.3) | 72.2 (0.0) | n/a |
| $A_4^m C_4^m$ | 86.4 (0.0) | 77.3 (0.0) | 72.7 (0.0) | 50.0 (0.0) | 82.4 (0.0) | 70.6 (0.0) | 47.1 (0.0) | n/a |
| $A_4^m D_4^m$ | 85.7 (0.0) | 76.2 (0.0) | 47.6 (0.0) | 47.6 (0.0) | 79.4 (3.4) | 58.8 (0.0) | 47.1 (0.0) | n/a |
| $A_5^m B_5^m$ | 91.3 (0.0) | 87.0 (0.0) | 60.9 (8.7) | 56.5 (0.0) | 94.4 (0.0) | 77.8 (0.0) | 55.6 (0.0) | n/a |
| $A_5^m C_5^m$ | 85.0 (0.0) | 80.0 (0.0) | 55.0 (0.0) | 55.0 (0.0) | 80.0 (0.0) | 73.3 (0.0) | 53.3 (0.0) | n/a |
| $A_5^m D_5^m$ | 84.2 (0.0) | 68.4 (0.0) | 52.6 (0.0) | n/a | 80.0 (0.0) | 53.3 (0.0) | 53.3 (0.0) | n/a |

**Table B.12.:** List of evaluation results for two concatenated middle parts of letter trajectories using the combinded UKR/CLM system for $N^{th} = 3, 4$. This table constitutes the extension of Table 6.6 in Section 6.3.7. The values are: mean (stdvar).

| | $N^{th} = 1$ | | | | $N^{th} = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $B_1^m A_1^m$ | 77.6 (0.7) | 94.9 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 92.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_1^m C_1^m$ | 88.9 (0.0) | 88.9 (0.0) | 88.9 (0.0) | 98.6 (0.0) | 75.7 (0.0) | 95.3 (1.4) | 100.0 (0.0) | 91.9 (0.0) |
| $B_1^m D_1^m$ | 92.4 (0.7) | 93.4 (0.0) | 90.8 (0.0) | 88.2 (0.0) | 97.4 (0.0) | 89.1 (1.3) | 82.1 (0.0) | 76.9 (0.0) |
| $B_2^m A_2^m$ | 87.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 92.5 (0.0) | 92.5 (0.0) | 82.5 (0.0) | 77.5 (0.0) |
| $B_2^m C_2^m$ | 85.4 (0.0) | 84.1 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 83.3 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_2^m D_2^m$ | 85.9 (0.0) | 86.2 (0.6) | 92.3 (0.0) | 88.5 (0.0) | 82.5 (0.0) | 87.5 (0.0) | 82.5 (0.0) | 80.0 (0.0) |
| $B_3^m A_3^m$ | 87.5 (0.8) | 100.0 (0.0) | 100.0 (0.0) | 97.8 (2.5) | 99.3 (1.4) | 100.0 (0.0) | 97.1 (0.0) | 71.4 (0.0) |
| $B_3^m C_3^m$ | 79.6 (9.3) | 93.4 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 91.0 (1.5) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_3^m D_3^m$ | 86.1 (0.7) | 92.9 (0.0) | 91.4 (0.0) | 87.1 (0.0) | 91.7 (0.0) | 88.9 (0.0) | 80.6 (0.0) | 75.0 (0.0) |
| $B_4^m A_4^m$ | 90.9 (0.0) | 92.4 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 97.1 (0.0) | 82.4 (0.0) |
| $B_4^m C_4^m$ | 96.1 (0.9) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_4^m D_4^m$ | 100.0 (0.0) | 91.9 (0.0) | 90.3 (0.0) | 85.5 (0.0) | 100.0 (0.0) | 93.8 (0.0) | 84.4 (0.0) | 81.3 (0.0) |
| $B_5^m A_5^m$ | 79.4 (0.0) | 79.4 (0.0) | 64.7 (0.0) | 60.3 (0.0) | 82.9 (0.0) | 60.0 (0.0) | 40.0 (0.0) | 40.0 (0.0) |
| $B_5^m C_5^m$ | 75.3 (0.7) | 69.4 (0.0) | 66.7 (0.0) | 62.5 (0.0) | 70.3 (0.0) | 62.2 (0.0) | 43.2 (0.0) | 43.2 (0.0) |
| $B_5^m D_5^m$ | 82.9 (0.0) | 68.6 (0.0) | 65.7 (0.0) | 61.4 (0.0) | 69.4 (0.0) | 61.1 (0.0) | 41.7 (0.0) | 41.7 (0.0) |
| $C_1^m A_1^m$ | 87.5 (0.8) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_1^m B_1^m$ | 94.8 (0.7) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 56.8 (0.0) |
| $C_1^m D_1^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 52.9 (0.0) |
| $C_2^m A_2^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 94.4 (0.0) | 82.6 (5.7) |
| $C_2^m B_2^m$ | 86.6 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 88.1 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_2^m D_2^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 47.2 (0.0) |
| $C_3^m A_3^m$ | 98.2 (3.6) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 97.2 (0.0) |
| $C_3^m B_3^m$ | 78.9 (9.2) | 96.1 (0.0) | 98.7 (0.0) | 100.0 (0.0) | 97.4 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 97.4 (0.0) |
| $C_3^m D_3^m$ | 95.1 (9.7) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 45.9 (0.0) |
| $C_4^m A_4^m$ | 90.6 (0.0) | 90.6 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_4^m B_4^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 51.5 (0.0) |
| $C_4^m D_4^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 48.4 (0.0) |
| $C_5^m A_5^m$ | 93.1 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 98.3 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $C_5^m B_5^m$ | 86.1 (0.0) | 84.7 (0.0) | 66.7 (0.0) | 62.5 (0.0) | 86.5 (0.0) | 52.7 (10.9) | 43.2 (0.0) | 43.2 (0.0) |
| $C_5^m D_5^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 48.4 (0.0) | 48.4 (0.0) |
| $D_1^m A_1^m$ | 87.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_1^m B_1^m$ | 93.4 (0.0) | 98.0 (0.8) | 100.0 (0.0) | 100.0 (0.0) | 97.4 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_1^m C_1^m$ | 98.5 (0.0) | 97.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 97.1 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_2^m A_2^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 96.3 (7.4) |
| $D_2^m B_2^m$ | 85.9 (0.0) | 97.4 (0.0) | 97.4 (0.0) | 97.4 (0.0) | 90.0 (0.0) | 97.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_2^m C_2^m$ | 98.6 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_3^m A_3^m$ | 96.1 (4.5) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_3^m B_3^m$ | 87.1 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 97.2 (0.0) | 97.2 (0.0) | 97.2 (0.0) | 97.2 (0.0) |
| $D_3^m C_3^m$ | 90.6 (11.0) | 98.6 (0.0) | 97.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_4^m A_4^m$ | 90.3 (0.0) | 90.3 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_4^m B_4^m$ | 98.4 (0.0) | 96.8 (0.0) | 96.8 (0.0) | 100.0 (0.0) | 96.9 (0.0) | 100.0 (0.0) | 96.9 (0.0) | 96.9 (0.0) |
| $D_4^m C_4^m$ | 97.1 (0.8) | 95.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 96.8 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $D_5^m A_5^m$ | 92.9 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 96.6 (0.0) | 100.0 (0.0) | 89.7 (0.0) |
| $D_5^m B_5^m$ | 84.3 (0.0) | 78.6 (0.0) | 65.7 (0.0) | 61.4 (0.0) | 83.3 (0.0) | 61.1 (0.0) | 41.7 (0.0) | 41.7 (0.0) |
| $D_5^m C_5^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |

**Table B.13.:** List of evaluation results for two concatenated middle parts of letter trajectories using the combinded UKR/CLM system for $N^{th} = 3, 4$. This table constitutes the extension of Table 6.6 in Section 6.3.7. The values are: mean (stdvar).

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $B_1^m A_1^m$ | 100.0 (0.0) | 100.0 (0.0) | 87.0 (10.7) | 33.3 (0.0) | 98.8 (2.4) | 100.0 (0.0) | 64.3 (41.2) | 28.6 (0.0) |
| $B_1^m C_1^m$ | 92.0 (0.0) | 100.0 (0.0) | 88.0 (0.0) | 100.0 (0.0) | 96.1 (2.6) | 96.1 (2.6) | 94.7 (0.0) | n/a |
| $B_1^m D_1^m$ | 92.3 (0.0) | 76.9 (0.0) | 73.1 (0.0) | 65.4 (0.0) | 90.0 (0.0) | 70.0 (0.0) | 70.0 (0.0) | 45.0 (0.0) |
| $B_2^m A_2^m$ | 100.0 (0.0) | 84.6 (0.0) | 42.3 (0.0) | 34.6 (0.0) | 100.0 (0.0) | 76.2 (0.0) | 38.1 (0.0) | 28.6 (0.0) |
| $B_2^m C_2^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) |
| $B_2^m D_2^m$ | 84.6 (0.0) | 88.5 (0.0) | 80.8 (0.0) | 57.7 (0.0) | 85.7 (0.0) | 81.0 (0.0) | 76.2 (0.0) | 100.0 (0.0) |
| $B_3^m A_3^m$ | 100.0 (0.0) | 100.0 (0.0) | 41.7 (0.0) | 25.0 (0.0) | 44.4 (0.0) | 44.4 (0.0) | 38.9 (0.0) | n/a |
| $B_3^m C_3^m$ | 92.3 (15.4) | 100.0 (0.0) | 88.5 (0.0) | 88.5 (0.0) | 50.0 (0.0) | 50.0 (0.0) | 50.0 (0.0) | 50.0 (0.0) |
| $B_3^m D_3^m$ | 95.8 (0.0) | 83.3 (0.0) | 75.0 (0.0) | 45.8 (0.0) | 47.4 (0.0) | 47.4 (0.0) | 47.4 (0.0) | n/a |
| $B_4^m A_4^m$ | 95.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 27.3 (0.0) | 100.0 (0.0) | 70.8 (2.8) | 38.9 (0.0) | n/a |
| $B_4^m C_4^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $B_4^m D_4^m$ | 95.2 (0.0) | 85.7 (0.0) | 90.5 (0.0) | 52.4 (0.0) | 94.1 (0.0) | 82.4 (0.0) | 91.2 (3.4) | n/a |
| $B_5^m A_5^m$ | 95.7 (0.0) | 91.3 (0.0) | 87.0 (0.0) | 95.7 (0.0) | 94.4 (0.0) | 73.6 (2.8) | 94.4 (0.0) | n/a |
| $B_5^m C_5^m$ | 79.0 (10.0) | 92.0 (0.0) | 90.0 (4.0) | 96.0 (0.0) | 53.9 (2.6) | 88.2 (2.6) | 89.5 (0.0) | n/a |
| $B_5^m D_5^m$ | 79.2 (0.0) | 83.3 (0.0) | 75.0 (0.0) | 84.4 (2.1) | 78.9 (0.0) | 78.9 (0.0) | 89.5 (0.0) | n/a |
| $C_1^m A_1^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $C_1^m B_1^m$ | 100.0 (0.0) | 100.0 (0.0) | 56.0 (0.0) | 56.0 (0.0) | 100.0 (0.0) | 57.9 (0.0) | 57.9 (0.0) | n/a |
| $C_1^m D_1^m$ | 100.0 (0.0) | 100.0 (0.0) | 52.2 (0.0) | 52.2 (0.0) | 100.0 (0.0) | 52.9 (0.0) | 52.9 (0.0) | n/a |
| $C_2^m A_2^m$ | 100.0 (0.0) | 91.7 (0.0) | 83.3 (0.0) | 46.9 (14.6) | 100.0 (0.0) | 100.0 (0.0) | 94.7 (0.0) | n/a |
| $C_2^m B_2^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 44.6 (25.0) | 100.0 (0.0) | 100.0 (0.0) | 54.5 (0.0) | 42.0 (11.4) |
| $C_2^m D_2^m$ | 100.0 (0.0) | 100.0 (0.0) | 45.8 (0.0) | 45.8 (0.0) | 100.0 (0.0) | 47.4 (0.0) | 47.4 (0.0) | n/a |
| $C_3^m A_3^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 54.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 55.6 (0.0) | n/a |
| $C_3^m B_3^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 30.8 (0.0) | 50.0 (0.0) | 50.0 (0.0) | 50.0 (0.0) | 0.0 (0.0) |
| $C_3^m D_3^m$ | 100.0 (0.0) | 100.0 (0.0) | 45.8 (0.0) | 45.8 (0.0) | 100.0 (0.0) | 47.4 (0.0) | 47.4 (0.0) | n/a |
| $C_4^m A_4^m$ | 95.5 (0.0) | 100.0 (0.0) | 95.5 (0.0) | 50.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 88.2 (0.0) | n/a |
| $C_4^m B_4^m$ | 100.0 (0.0) | 100.0 (0.0) | 50.0 (0.0) | 62.5 (35.9) | 100.0 (0.0) | 52.9 (0.0) | 50.0 (5.9) | n/a |
| $C_4^m D_4^m$ | 100.0 (0.0) | 86.9 (26.2) | 47.6 (0.0) | 47.6 (0.0) | 100.0 (0.0) | 50.0 (0.0) | 50.0 (0.0) | n/a |
| $C_5^m A_5^m$ | 100.0 (0.0) | 100.0 (0.0) | 85.0 (0.0) | 55.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 53.3 (0.0) | n/a |
| $C_5^m B_5^m$ | 96.0 (0.0) | 96.0 (0.0) | 52.0 (0.0) | 52.0 (0.0) | 86.8 (15.8) | 52.6 (0.0) | 52.6 (0.0) | n/a |
| $C_5^m D_5^m$ | 100.0 (0.0) | 47.6 (0.0) | 47.6 (0.0) | 47.6 (0.0) | 100.0 (0.0) | 50.0 (0.0) | 50.0 (0.0) | n/a |
| $D_1^m A_1^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 36.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 48.7 (34.2) | n/a |
| $D_1^m B_1^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 73.1 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 55.0 (0.0) | 45.0 (0.0) |
| $D_1^m C_1^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_2^m A_2^m$ | 100.0 (0.0) | 100.0 (0.0) | 59.1 (0.0) | 40.9 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_2^m B_2^m$ | 96.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 57.7 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 66.7 (0.0) | 57.1 (0.0) |
| $D_2^m C_2^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_3^m A_3^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 94.1 (0.0) | n/a |
| $D_3^m B_3^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 64.6 (7.2) | 47.4 (0.0) | 42.1 (0.0) | 42.1 (0.0) | n/a |
| $D_3^m C_3^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_4^m A_4^m$ | 95.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 92.9 (2.7) | 100.0 (0.0) | 94.1 (0.0) | 35.3 (0.0) | n/a |
| $D_4^m B_4^m$ | 95.2 (0.0) | 95.2 (0.0) | 95.2 (0.0) | 52.4 (0.0) | 94.1 (0.0) | 94.1 (0.0) | 58.8 (0.0) | n/a |
| $D_4^m C_4^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_5^m A_5^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a | 100.0 (0.0) | 100.0 (0.0) | 50.0 (33.3) | n/a |
| $D_5^m B_5^m$ | 95.8 (0.0) | 95.8 (0.0) | 91.7 (0.0) | 58.3 (0.0) | 94.7 (0.0) | 89.5 (0.0) | 57.9 (0.0) | n/a |
| $D_5^m C_5^m$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 72.6 (2.4) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |

**Table B.14.:** List of evaluation results for two concatenated middle parts of letter trajectories using the combinded UKR/CLM system for $N^{th} = 3, 4$. This table constitutes the extension of Table 6.6 in Section 6.3.7. The values are: mean (stdvar).

### B.2.4. Single whole letter trajectories ("ABD" CLM & weak background)

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_5$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a | 100.0 (0.0) | 69.2 (0.0) | n/a | n/a |
| $B_1$ | 84.6 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 88.5 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 90.0 (0.0) | 90.0 (0.0) |
| $B_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 94.2 (1.7) | 100.0 (0.0) | 100.0 (0.0) | 95.5 (0.0) | 81.8 (0.0) |
| $B_3$ | 93.8 (12.5) | 100.0 (0.0) | 100.0 (0.0) | 79.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $B_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $B_5$ | 86.5 (7.4) | 96.2 (0.0) | 96.2 (0.0) | 88.5 (0.0) | 95.0 (0.0) | 95.0 (0.0) | 95.0 (0.0) | 90.0 (0.0) |
| $C_1$ | 0.0 (0.0) | 0.0 (0.0) | 25.0 (0.0) | 25.0 (0.0) | 0.0 (0.0) | 26.7 (0.0) | 26.7 (0.0) | n/a |
| $C_2$ | 2.1 (4.2) | 0.0 (0.0) | 38.5 (2.1) | 47.9 (4.2) | 0.0 (0.0) | 16.7 (0.0) | 44.4 (4.5) | n/a |
| $C_3$ | 0.0 (0.0) | 4.0 (0.0) | 16.0 (0.0) | 25.0 (2.0) | 0.0 (0.0) | 15.8 (0.0) | 19.7 (2.6) | n/a |
| $C_4$ | 0.0 (0.0) | 15.0 (0.0) | 20.0 (0.0) | 20.0 (0.0) | 0.0 (0.0) | 20.0 (0.0) | 20.0 (0.0) | n/a |
| $C_5$ | 0.0 (0.0) | 15.0 (0.0) | 15.0 (0.0) | 15.0 (0.0) | 0.0 (0.0) | 13.3 (0.0) | 18.3 (3.3) | n/a |
| $D_1$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a | 100.0 (0.0) | 100.0 (0.0) | n/a | n/a |
| $D_5$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a | 100.0 (0.0) | 100.0 (0.0) | n/a | n/a |

**Table B.15.:** List of evaluation results for single whole letter trajectories using the combinded UKR/CLM system with only four layers for the letters '$A$', '$B$' and '$D$' plus background. Depicted are the evaluation results for CLM with *weak* background layer, i.e. $\lambda_{bg} = (1 - 10/N)$ (for $N > 10$) and $K_{bg} = H$, for the sampling rates $N^{th} = 3, 4$. This table constitutes the extension of Table 6.8 in Section 6.3.9. The values are: mean (stdvar).

## B.2.5. Single whole letter trajectories ("ABD" CLM & strong background)

| | $N^{th} = 3$ | | | | $N^{th} = 4$ | | | |
|---|---|---|---|---|---|---|---|---|
| Seq | H 5 | H 10 | H 15 | H 20 | H 5 | H 10 | H 15 | H 20 |
| $A_1$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $A_5$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a | 100.0 (0.0) | 69.2 (0.0) | n/a | n/a |
| $B_1$ | 42.3 (31.1) | 100.0 (0.0) | 98.1 (2.2) | 88.5 (0.0) | 92.5 (5.0) | 100.0 (0.0) | 90.0 (0.0) | 88.8 (2.5) |
| $B_2$ | 39.2 (1.7) | 100.0 (0.0) | 100.0 (0.0) | 21.7 (43.3) | 62.5 (15.0) | 100.0 (0.0) | 68.2 (45.6) | 0.0 (0.0) |
| $B_3$ | 77.1 (15.4) | 100.0 (0.0) | 87.5 (3.4) | 79.2 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $B_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $B_5$ | 38.5 (0.0) | 96.2 (0.0) | 96.2 (0.0) | 88.5 (0.0) | 92.5 (5.0) | 95.0 (0.0) | 95.0 (0.0) | 90.0 (0.0) |
| $C_1$ | 0.0 (0.0) | 0.0 (0.0) | 25.0 (0.0) | 25.0 (0.0) | 0.0 (0.0) | 26.7 (0.0) | 26.7 (0.0) | n/a |
| $C_2$ | 77.1 (26.5) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 50.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $C_3$ | 75.0 (29.3) | 78.0 (25.4) | 100.0 (0.0) | 100.0 (0.0) | 44.7 (3.0) | 25.0 (18.4) | 100.0 (0.0) | n/a |
| $C_4$ | 75.0 (28.9) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 0.0 (0.0) | 20.0 (0.0) | 20.0 (0.0) | n/a |
| $C_5$ | 32.5 (21.8) | 57.5 (49.1) | 100.0 (0.0) | 100.0 (0.0) | 18.3 (21.3) | 13.3 (0.0) | 60.0 (46.2) | n/a |
| $D_1$ | 75.0 (50.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_2$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_3$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a |
| $D_4$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a | 100.0 (0.0) | 100.0 (0.0) | n/a | n/a |
| $D_5$ | 100.0 (0.0) | 100.0 (0.0) | 100.0 (0.0) | n/a | 100.0 (0.0) | 100.0 (0.0) | n/a | n/a |

**Table B.16.:** List of evaluation results for single whole letter trajectories using the combinded UKR/CLM system with only four layers for the letters '$A$', '$B$' and '$D$' plus background. Depicted are the evaluation results for CLM with *strong* background layer, i.e. $\lambda_{bg}^{strong} = 3\lambda_{bg}$ and $K_{bg} = H$, for the sampling rates $N^{th} = 3, 4$. This table constitutes the extension of Table 6.9 in Section 6.3.9. The values are: mean (stdvar).

# References


Arsenio, A. M. (2004, September). Learning task sequences from scratch: applications to the control of tools and toys by a humanoid robot. In *Proceedings of the Int. Conference on Control Applications* (p. 400-405). Taipei, Taiwan.

Atkeson, C. G., Moore, A. W., & Schaal, S. (1997a, February). Locally weighted learning. *Artificial Intelligence Review (Special issue on lazy learning)*, *11*(1-5), 11–73.

Atkeson, C. G., Moore, A. W., & Schaal, S. (1997b, February). Locally weighted learning for control. *Artificial Intelligence Review (Special issue on lazy learning)*, *11*(1-5), 75 – 113.

Atkeson, C. G., & Schaal, S. (1995, December). Memory-based neural networks for robot learning. *Neurocomputing*, *9*(3), 243 - 269. (Control and Robotics, Part III)

Atkeson, C. G., & Schaal, S. (1997, July). Robot Learning From Demonstration. In *Proceedings of the Int. Conference on Machine Learning (ICML).* Nashville, Tennessee, USA.

Barreto, G., Araújo, A., & H.Ritter. (2003, April). Self-Organizing Feature Maps for Modeling and Control of Robotic Manipulators. *Journal of Intelligent and Robotic Systems*, *36*(4), 407-450.

Bentivegna, D. C. (2004). *Learning from Observation Using Primitives.* PhD Thesis, College of Computing, Georgia Institute of Technology, Atlanta, GA, USA.

Bentivegna, D. C., Atkeson, C. G., & Cheng, G. (2004, June). Learning tasks from observation and practice. *Robotics and Autonomous Systems (Special issue on Robot Learning from Demonstration)*, *47*(2 – 3), 163 – 169.

Beth, T., Boesnach, I., Haimerl, M., Moldenhauer, J., Bös, K., & Wank, V. (2003, October). Characteristics in Human Motion – From Acquisition to Analysis. In *Proceedings of the Int. Conference on Humanoid Robots (HUMANOIDS).* Karlsruhe and Munich, Germany.

Bishop, C. M., & Legleye, C. (1995, July). Estimating conditional probability densities for periodic variables. In *Advances in Neural Information Processing Systems 7 (Proceedings of the Int. Conference on Neural Information Processing Systems (NIPS), 1994)* (pp. 641–648). MIT Press, Cambridge, MA, USA.

Bitzer, S., Havoutis, I., & Vijayakumar, S. (2008, July). Synthesising Novel Movements

through Latent Space Modulation of Scalable Control Policies. In *From Animals to Animats 10 (Proceedings of the 10th Int. Conference on Simulation of Adaptive Behaviour (SAB))* (pp. 199 – 209). Osaka, Japan: Springer Berlin/Heidelberg.

Bitzer, S., & Vijayakumar, S. (2009, December). Latent Spaces for Dynamic Movement Primitives. In *Proceedings of the Int. Conference on Humanoid Robots (HUMANOIDS)* (pp. 574 – 581). Paris, France.

Borst, C., Fischer, M., & Hirzinger, G. (2002, September). Calculating hand configurations for precision and pinch grasps. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)* (pp. 1553 – 1559). Lausanne, Switzerland.

Borst, C., Fischer, M., & Hirzinger, G. (2003, October). Grasping the Dice by Dicing the Grasp. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)* (pp. 3692 – 3697). Las Vegas, USA.

Breazeal, C., & Scassellati, B. (2002, November). Robots that imitate humans. *Trends in Cognitive Sciences*, *6*(11), 481 - 487.

Butterfass, J., Grebenstein, M., Liu, H., & Hirzinger, G. (2001, May). DLR-Hand II: next generation of a dextrous robot hand. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 109 – 114). Seoul, Korea.

Calinon, S., & Billard, A. (2004, September). Stochastic Gesture Production and Recognition Model for a Humanoid Robot. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)* (pp. 2769 – 2774). Sendai, Japan.

Calinon, S., Guenter, F., & Billard, A. (2006, May). On Learning the Statistical Representation of a Task and Generalizing it to Various Contexts. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 2978–2983). Orlando, FL, USA.

Calinon, S., Guenter, F., & Billard, A. (2007). On Learning, Representing and Generalizing a Task in a Humanoid Robot. *IEEE Transactions on Systems, Man and Cybernetics, Part B. (Special issue on robot learning by observation, demonstration and imitation)*, *37*(2), 286–298.

Chang, L., Pollard, N., Mitchell, T., & Xing, E. (2007, October). Feature Selection for Grasp Recognition from Optical Markers. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)* (pp. 2944 – 2950). San Diego, USA.

Chernova, S., & Veloso, M. (2007, May). Confidence-Based Policy Learning from Demonstration Using Gaussian Mixture Models. In *Proceedings of the Int. Conference on Autonomous Agents and Multiagent Systems (AAMAS)* (pp. 1177 – 1184). Honolulu, Hawaii, USA.

Chou, T.-S., Gadd, A., & Knott, D. (2000, April). Hand-Eye: A Vision-Based Approach to Data Glove Calibration. In *Proceedings of the Human Interface Technologies 2000*

*Conference* (pp. 47 – 54). Vancouver, Canada.

Ciocarlie, M., Goldfeder, C., & Allen, P. (2007, October). Dimensionality Reduction for Hand-Independent Dexterous Robotic Grasping. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)* (pp. 3270 – 3275). San Diego, USA.

CM-Labs. (2005). *Vortex 2.1.* www.cm-labs.com/products/vortex.

Cox, T. F., & Cox, M. A. A. (2000). *Multidimensional Scaling* (2nd ed.). Boca Raton, USA: Chapman & Hall / CRC Press.

Demiris, J., & Hayes, G. R. (2002, May). Imitation as a dual-route process featuring predictive and learning components: a biologically-plausible computational model. In K. Dautenhahn & C. Nehaniv (Eds.), *Imitation in Animals and Artifacts* (pp. 327 – 361). MIT Press, Cambridge, MA, USA.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B (Methodological)*, *39*(1), 1–38.

Dijkstra, E. W. (1959, December). A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, *1*(1), 269 – 271.

Ehrenmann, M., Zöllner, R., Rogalla, O., Vacek, S., & Dillmann, R. (2003, October). Observation in Programming by Demonstration: Training and Execution Environment. In *Proceedings of the Int. Conference on Humanoid Robots (HUMANOIDS).* Karlsruhe and Munich, Germany.

Ek, C. H., Rihan, J., Torr, P., Rogez, G., & Lawrence, N. D. (2008, September). Ambiguity modeling in latent spaces. In *Machine Learning for Multimodal Interaction (Proceedings of the 5th Int. Workshop on Machine Learning for Multimodal Interaction (MLMI))* (pp. 62–73). Utrecht, The Netherlands: Springer Berlin/Heidelberg.

Ek, C. H., Torr, P., & Lawrence, N. D. (2008, September). Gaussian Process Latent Variable Models for Human Pose Estimation. In *Machine Learning for Multimodal Interaction (Proceedings of the 5th Int. Workshop on Machine Learning for Multimodal Interaction (MLMI))* (pp. 132–143). Utrecht, The Netherlands: Springer Berlin/Heidelberg.

Falck-Ytter, T., Gredebäck, G., & von Hofsten, C. (2006, July). Infants predict other people's action goals. *Nature Neuroscience*, *9*(7), 878 – 879.

Fischer, M., Smagt, P. van der, & Hirzinger, G. (1998, May). Learning Techniques in a Dataglove Based Telemanipulation System for the DLR Hand. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 1603 – 1608). Leuven, Belgium.

Fuentes, O., & Nelson, R. (1998, April). Learning Dextrous Manipulation Skills for Multifingered Robot Hands Using the Evolution Strategy. *Machine Learning*, *31*, 223 –

237.

Griffin, W., Findley, R., Turner, M., & Cutkosky, M. (2000, November). Calibration and Mapping of a Human Hand for Dexterous Telemanipulation. In *Proceedings of the ASME International Mechanical Engineering Congress & Exposition (IMECE), "Haptic Interfaces for Virtual Environments and Teleoperator Systems" Symposium.* Orlando, USA.

Grochow, K., Martin, S. L., Hertzmann, A., & Popovic, Z. (2004, August). Style-based inverse kinematics. In *Proceedings of the Int. Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)* (pp. 522 – 531).

Han, L., Li, Z., Trinkle, J., Qin, Z., & Jiang, S. (2000, April). The Planning and Control of Robot Dextrous Manipulation. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 263 – 269). San Francisco, USA.

Härdle, W., & Marron, J. S. (1985, December). Optimal Bandwidth Selection in Nonparametric Regression Function Estimation. *The Annals of Statistics*, *13*(4), 1465 – 1481.

Haschke, R., Steil, J. J., Steuwer, I., & Ritter, H. (2005, June). Task-Oriented Quality Measures for Dextrous Grasping. In *Proceedings of the Int. Symposium on Computational Intelligence in Robotics and Automation (CIRA)* (pp. 689 – 694). Espoo, Finland.

Hollister, A., Buford, W. L., Myers, L. M., Giurintano, D. J., & Novick, A. (1992, February). The axes of rotation of the thumb carpometacarpal joint. *Journal of Orthopaedic Research*, *10*(3), 454 – 460.

Hollister, A., Giurintano, D. J., Buford, W. L., Myers, L. M., & Novick, A. (1995, November). The axes of rotation of the thumb interphalangeal and metacarpophalangeal joints. *Clinical Orthopaedics & Related Research*(320), 188 – 193.

Hou, S., Galata, A., Caillette, F., Thacker, N., & Bromiley, P. (2007, October). Real-time Body Tracking Using a Gaussian Process Latent Variable Model. In *Proceedings of the Int. Conference on Computer Vision (ICCV)* (pp. 1–8). Rio de Janeiro, Brazil.

Iba, S., Paredis, C. J. J., & Khosla, P. (2005, January). Interactive Multimodal Robot Programming. *The International Journal of Robotics Research*, *24*(1), 83–104.

Igel, C., & Hüsken, M. (2000, May). Improving the RPROP Learning Algorithm. In H. Bothe & R. Rojas (Eds.), *Proceedings of the* $2^{nd}$ *Int. ICSC Symposium on Neural Computation (NC 2000)* (pp. 115 – 121). Berlin, Germany: ICSC Academic Press.

Ijspeert, A., Nakanishi, J., & Schaal, S. (2003, October). Learning attractor landscapes for learning motor primitives. In S. Becker, S. Thrun, & K. Obermayer (Eds.), *Advances in neural information processing systems 16 (proceedings of the int. conference on neural information processing systems (nips), 2002)* (pp. 1523–1530). MIT Press,

Cambridge, MA, USA.

Jacobsen, S., Iversen, E., Knutti, D., Johnson, R., & Biggers, K. (1986, April). Design of the Utah/M.I.T. Dextrous Hand. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 1520 – 1532). Washington DC, USA.

Kahlesz, F., Zachmann, G., & Klein, R. (2004, June). Visual-fidelity dataglove calibration. In *Proceedings of the Computer Graphics International (CGI)* (pp. 403–410). Crete, Greece.

Keogh, E., & Pazzani, M. J. (2001, April). Derivative Dynamic Time Warping. In *First SIAM International Conference on Data Mining.* Chicago, IL, USA.

Klanke, S. (2007). *Learning Manifolds with the Parametrized Self-Organizing Map and Unsupervised Kernel Regression.* PhD Thesis, Bielefeld University, Bielefeld, Germany.

Klanke, S., & Ritter, H. (2007, March). Variants of Unsupervised Kernel Regression: General Cost Functions. *Neurocomputing (Advances in Computational Intelligence and Learning - 14th European Symposium on Artificial Neural Networks 2006)*, *70*(7 – 9), 1289 – 1303.

Klanke, S., & Ritter, H. J. (2006, Septempber). A Leave-K-Out Cross-Validation Scheme for Unsupervised Kernel Regression. In S. Kollias, A. Stafylopatis, W. Duch, & E. Oja (Eds.), *Artificial Neural Networks – ICANN 2006* (Vol. 4132, pp. 427 – 436). Springer Berlin/Heidelberg.

Lawrence, N. D. (2004, July). Gaussian process models for visualisation of high dimensional data. In *Advances in Neural Information Processing Systems (Proceedings of the Int. Conference on Neural Information Processing Systems (NIPS), 2004)* (pp. 329 – 336). MIT Press, Cambridge, MA, USA.

Lawrence, N. D. (2005). Probabilistic Non-linear Principal Component Analysis with Gaussian Process Latent Variable Models. *Journal of Machine Learning Research*, *6*, 1783 – 1816.

Lawrence, N. D., & Moore, A. J. (2007). Hierarchical Gaussian process latent variable models. In *Proceedings of the Int. Conference in Machine Learning (ICML)* (pp. 481 – 488). Corvalis, Oregon, USA.

Lawrence, N. D., & Quiñonero Candela, J. (2006). Local distance preservation in the GP-LVM through back constraints. In *Proceedings of the Int. Conference in Machine Learning (ICML)* (pp. 513–520). Pittsburgh, Pennsylvania, USA.

Mardia, K. V. (1972). *Statistics of Directional Data.* Academic Press, London.

McLachlan, G., & Peel, D. (2000). *Finite Mixture Models* (1st ed.). New York, USA: Wiley.

Meinicke, P. (2000). *Unsupervised Learning in a Generalized Regression Framework.* PhD

Thesis, Bielefeld University, Bielefeld, Germany.

Meinicke, P., Klanke, S., Memisevic, R., & Ritter, H. (2005). Principal Surfaces from Unsupervised Kernel Regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, *27*(9), 1379 – 1391.

Michelman, P., & Allen, P. (1994, May). Forming Complex Dextrous Manipulations from Task Primitives. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 3383–3388). San Diego, USA.

Miller, A., & Allen, P. (2000). GraspIt!: A Versatile Simulator for Grasp Analysis. In *Proceedings of the ASME International Mechanical Engineering Congress & Exposition (IMECE)* (pp. 1251 – 1258). Orland, USA.

Miller, A., & Allen, P. K. (2004, December). Graspit!: A Versatile Simulator for Robotic Grasping. *IEEE Robotics and Automation Magazine*, *11*(4), 110 – 122.

Miller, A., Knoop, S., Allen, P., & Christensen, H. (2003). Automatic grasp planning using shape primitives. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA).*

Møller, M. F. (1993). A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, *6*, 525–533.

Mouri, T., Kawasaki, H., Yoshikawa, K., Takai, J., & Ito, S. (2002, October). Anthropomorphic Robot Hand: Gifu Hand III. In *Proceedings of the Int. Conference on Control, Automation and Systems (ICCAS)* (pp. 1288 – 1293). Jeonbuk, Korea.

Mühlig, M., Gienger, M., Hellbach, S., Steil, J. J., & Goerick, C. (2009, October). Task-level Imitation Learning using Variance-based Movement Optimization. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)* (pp. 4996 – 5002). St. Louis, USA.

Mühlig, M., Gienger, M., Steil, J. J., & Goerick, C. (2009, May). Automatic Selection of Task Spaces for Imitation Learning. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 1177 – 1184). Kobe, Japan.

Murray, R. M., Li, Z., & Sastry, S. S. (1994, March). A Mathematical Introduction to Robotic Manipulation. In (chap. Multifingered Hands and Dextrous Manipulation). CRC Press.

Nadaraya, E. A. (1964, January). On Estimating Regression. *Theory of Probability and Its Applications*, *9*, 141 – 142.

Natale, L., & Torres-Jara, E. (2006, September). A sensitive approach to grasping. In *6th Int. Conference on Epigenetic Robotics (EpiRob).* Paris, France.

Nattkemper, T. W., Wersing, H., Ritter, H., & Schubert, W. (2002, November). A Neu-

ral Network Architecture for Automatic Segmentation of Fluorescence Micrographs. *Neurocomputing*, *48*(4), 357-367.

Nicolescu, M. N., & Mataric, M. J. (2001, September). Learning and interacting in human-robot domains. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, *31*(5), 419–430.

Ontrup, J., & Ritter, H. (1998). *Perceptual Grouping in a Neural Model: Reproducing Human Texture Perception* (Tech. Rep. No. TR 98/6). Bielefeld, Germany: SFB 360, Bielefeld University.

Ontrup, J., Wersing, H., & Ritter, H. (2004, March). A Computational Feature Binding Model of Human Texture Perception. *Cognitive Processing*, *5*(1), 32 – 44.

Oztop, E., Hale, J., Babic, J., & Kawato, M. (2008, September). Robots as complex tools for humans to control: Human visuo-motor learning for robot skill synthesis. In *Proceedings of the Intelligent Robots and Systems (IROS) Workshop on Grasp and Task Learning by Imitation* (pp. 25 – 29). Nice, France.

Oztop, E., Lin, L.-H., Kawato, M., & Cheng, G. (2006, December). Dexterous Skills Transfer by Extending Human Body Schema to a Robotic Hand. In *Proceedings of the Int. Conference on Humanoid Robots (HUMANOIDS)* (pp. 82 – 87). Genova, Italy.

Oztop, E., Lin, L.-H., Kawato, M., & Cheng, G. (2007, April). Extensive Human Training for Robot Skill Synthesis: Validation on a Robotic Hand. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 1788 – 1793). Roma, Italy.

Pardowitz, M., Haschke, R., Steil, J., & Ritter, H. (2008, December). Gestalt-Based Action Segmentation for Robot Task Learning. In *Proceedings of the Int. Conference on Humanoid Robots (HUMANOIDS).* Daejeon, Korea.

Pardowitz, M., Steffen, J., & Ritter, H. (2009, September). Self-emerging Action Gestalts for Task Segmentation. In *Proceedings of the 32nd German Conference on Artificial Intelligence (KI 2009)* (pp. 589 – 596). Paderborn, Germany: Springer Berlin / Heidelberg.

Pelossof, R., Miller, A., Allen, P., & Jebara, T. (2004, April). An SVM Learning Approach to Robotic Grasping. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 3512 – 3518). New Orleans, USA.

Platt, R., Fagg, A., & Grupen, R. (2002, September). Nullspace Composition of Control Laws for Grasping. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)* (pp. 1717 – 1723). Lausanne, Switzerland.

Platt, R., Fagg, A., & Grupen, R. (2004, April). Manipulation Gaits: Sequences of Grasp Control Tasks. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 801 – 806). New Orleans, USA.

Pollard, N., & Hodgins, J. (2003, September). Generalizing Demonstrated Manipulation Tasks. In *Algorithmic Foundation of Robotics V (Proceedings of the Workshop on the Algorithmic Foundations of Robotics 2002 (WAFR '02)* (pp. 523 – 540). Springer, Berlin.

Ratanamahatana, C. A., & Keogh, E. (2004, August). Everything you know about dynamic time warping is wrong. In *3rd Workshop on Mining Temporal and Sequential Data, in conjunction with 10th ACM SIGKDD Int. Conference on Knowledge Discovery and Data Mining (KDD-2004).* Seattle, WA, USA.

Reichel, M., & The Shadow Robot Company. (2004, July). Transformation of Shadow Dextrous Hand and Shadow Finger Test Unit from Prototype to Product for Intelligent Manipulation and Grasping. In *Proceedings of Int. Conference on Intelligent Manipulation and Grasping (IMG)* (pp. 123 – 124). Genova, Italy.

Riedmiller, M., & Braun, H. (1993, March). A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP algorithm. In *Proceedings of the Int. Conference on Neural Networks (ICNN)* (pp. 586 – 591). San Francisco, USA.

Ritter, H. (1990, July). A Spatial Approach to Feature Linking. In *Proc. International Neural Network Conference (INNC).* Paris, France.

Ritter, H. (1990 – 2010). *Neo/NST: The Graphical Simulation Toolkit.* http://ni.www.techfak.uni-bielefeld.de/neo.

Ritter, H. (1993, September). Parametrized Self-Organizing Maps. In S. Gielen & B. Kappen (Eds.), *Proceedings of the Int. Conference on Artificial Neural Networks (ICANN)* (pp. 568 – 575). Springer London, UK.

Ritter, H., Haschke, R., & Steil, J. (2009). Trying to grasp a sketch of a brain for grasping. In B. Sendhoff, K. Edgar, O. Sporns, H. Ritter, & K. Doya (Eds.), *Creating Brain-Like Intelligence* (pp. 84 – 102). Springer Berlin/Heidelberg.

Ritter, H., Martinetz, T., & Schulten, K. (1992). *Neural Computation and Self-Organizing Maps: An Introduction.* Addison-Wesley.

Rizzolatti, G., & Craighero, L. (2004, March). The Mirror-Neuron System. *Annual Review of Neuroscience*, *27*, 169–192.

Rizzolatti, G., Fabbri-Destro, M., & Cattaneo, L. (2009, January). Mirror neurons and their clinical relevance. *Nature Clinical Practice Neurology*, *5*(1), 24–34.

Rosenblatt, M. (1971). Curve Estimates. *Annals of Mathematical Statistics*, *42*(6), 1815 – 1842.

Röthling, F., Haschke, R., Steil, J. J., & Ritter, H. (2007, October). Platform Portable Anthropomorphic Grasping with the Bielefeld 20-DOF Shadow and 9-DOF TUM Hand. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)*

(pp. 2951 – 2956). San Diego, USA.

Roweis, S. T., & Saul, L. K. (2000, December). Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science*, *290*(5500), 2323 – 2326.

Salisbury, J. K., Jr. (1985). Kinematic and force analysis of articulated hands. In *Recent advances in robotics* (pp. 131 – 174). New York, USA: John Wiley & Sons, Inc.

Schaal, S. (1997). Learning from Demonstration. In M. C. Mozer, M. I. Jordan, & T. Petsche (Eds.), *Advances in neural information processing systems* (Vol. 9, p. 1040-1046). The MIT Press.

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

Schaal, S., Ijspeert, A., & Billard, A. (2003). Computational approaches to motor learning by imitation. *Philosophical Transaction of The Royal Society of London: Series B, Biological Sciences*, *358*(1431), 537–547.

Schölkopf, B., Smola, A., & Müller, K. (1998, July). Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Computation*, *10*(5), 1299-1319.

Schwarz, G. (1978, March). Estimating the dimension of a model. *The Annals of Statistics*, *6*(2), 461 – 464.

Steffen, J. (2005). *Erfahrungsbasierte Greifstrategie für anthropomorphe Roboterhände.* Master Thesis (German *Diplomarbeit*), Bielefeld University, Germany. (German)

Steffen, J., Haschke, R., & Ritter, H. (2007a, October). Experience-based and Tactile-driven Dynamic Grasp Control. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)* (pp. 2938 – 2943). San Diego, USA.

Steffen, J., Haschke, R., & Ritter, H. (2007b, September). SOM-based Experience Representation for Dextrous Grasping. In *Proceedings of the $6^{th}$ International Workshop on Self-Organizing Maps (WSOM).* Bielefeld, Germany. (Best Paper Award)

Steffen, J., Haschke, R., & Ritter, H. (2008a, September). Towards Dextrous Manipulation Using Manifolds. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)* (pp. 2738 – 2743). Nice, France.

Steffen, J., Haschke, R., & Ritter, H. (2008b, June). Using Manifolds for Dextrous Hand Control. In *Procedings of the R:SS 2008 Workshop "Robot Manipulation: Intelligence in Human Environments", Robotics: Science and Systems Conference (R:SS).* Zurich, Switzerland.

Steffen, J., Klanke, S., Vijayakumar, S., & Ritter, H. (2009a, May). Realising Dextrous Manipulation with Structured Manifolds using Unsupervised Kernel Regression with Structural Hints. In *Proceedings of the Int. Conference on Robotics and Automation*

*(ICRA) 2009 Workshop: Approaches to Sensorimotor Learning on Humanoid Robots* (pp. 24 – 29). Kobe, Japan.

Steffen, J., Klanke, S., Vijayakumar, S., & Ritter, H. (2009b, June). Towards Semi-supervised Manifold Learning: UKR with Structural Hints. In *Proceedings of the International Workshop on Self-Organizing Maps (WSOM) 2009* (pp. 298 – 306). St.Augustine, USA: Springer Berlin / Heidelberg.

Steffen, J., Pardowitz, M., & Ritter, H. (2009a, September). A Manifold Representation as Common Basis for Action Production and Recognition. In *Proceedings of the 32nd German Conference on Artificial Intelligence (KI 2009)* (pp. 607 – 614). Paderborn, Germany: Springer Berlin / Heidelberg.

Steffen, J., Pardowitz, M., & Ritter, H. (2009b, October). Using Structured UKR Manifolds for Motion Classification and Segmentation. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)* (pp. 4785 – 4790). St.Louis, USA.

Steffen, J., Pardowitz, M., Steil, J., & Ritter, H. (2010, April). Neural competition for motion segmentation. In *Proceedings of the European Symposium on Artificial Neural Networks (ESANN).* Bruges, Belgium. (to appear)

Steil, J., Röthling, F., Haschke, R., & Ritter, H. (2004, June). Situated robot learning for multi-modal instruction and imitation of grasping. *Robotics and Autonomous Systems (Special Issue on Robot Learning by Demonstration)*, *47*(2 – 3), 129 – 141.

Suleiman, W., Yoshida, E., Kanehiro, F., Laumond, J.-P., & Monin, A. (2008, May). On Human Motion Imitation by Humanoid Robot. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 2697–2704). Pasadena, CA, USA.

Supuk, T., Kodek, T., & Bajd, T. (2005). Estimation of hand preshaping during human grasping. *Medical engineering & physics*, *27*(9), 790–797.

Tenenbaum, J. B., Silva, V. de, & Langford, J. C. (2000, December). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*(5500), 2319 – 2323.

Tsoli, A., & Jenkins, O. (2007, June). 2D subspaces for user-driven robot grasping. In *Robotics: Science and Systems (RSS) Workshop: "Robot Manipulation: Sensing and Adapting to the Real World".* Atlanta, USA.

Turner, M. (2001). *Programming Dexterous Manipulation by Demonstration.* PhD Thesis, Stanford University, Department of Mechanical Engineering, Stanford, USA.

Urtasun, R., Fleet, D. J., Geiger, A., Popovic, J., Darrell, T. J., & Lawrence, N. D. (2008, July). Topologically-constrained latent variable models. In *Proceedings of the Int. Conference in Machine Learning (ICML)* (pp. 1080 – 1087). Helsinki, Finland.

Vlassis, N., & Likas, A. (2002). A Greedy EM Algorithm for Gaussian Mixture Learning. *Neural Processing Letters*, *15*, 77–87.

Walter, J. (1996). *Rapid Learning in Robotics*. PhD Thesis, Bielefeld University, Faculty of Technology, Bielefeld, Germany.

Walter, J., & Ritter, H. (1996, July). Rapid Learning with Parametrized Self-organizing Maps. *Neurocomputing (Current European Neurocomputing Research)*, *12*(2-3), 131-153.

Watson, G. S. (1964, December). Smooth Regression Analysis. *Sankhya: The Indian Journal of Statistics Series A*, *26*(4), 359–372.

Weng, S., & Steil, J. (2002, August). Data Driven Generation of Interactions for Feature Binding and Relaxation Labeling. In *Proceedings of the Int. Conference on Artificial Neural Networks (ICANN)* (pp. 432 – 437). Madrid, Spain: Springer Berlin / Heidelberg.

Weng, S., & Steil, J. (2003, June). Learning Compatibility Functions for Feature Binding and Perceptual Grouping. In *Proceedings of the Int. Conference on Artificial Neural Networks (ICANN)* (pp. 60–67). Istanbul, Turkey: Springer Berlin / Heidelberg.

Weng, S., Wersing, H., Steil, J., & Ritter, H. (2006). Learning Lateral Interactions for Feature Binding and Sensory Segmentation from Prototypic Basis Interactions. *IEEE Transactions on Neural Networks*, *17*(4), 843-863.

Wersing, H. (2000). *Spatial Feature Binding and Learning in Competitive Neural Layer Architectures*. PhD Thesis, Faculty of Technology, Bielefeld University. (Göttingen, Cuvillier)

Wersing, H. (2001). Learning Lateral Interactions for Feature Binding and Sensory Segmentation. In *Advances in Neural Information Processing Systems 14 (Proceedings of the Int. Conference on Neural Information Processing Systems (NIPS), 2001).*

Wersing, H., Steil, J., & Ritter, H. (2001). A Competitive Layer Model for Feature Binding and Sensory Segmentation. *Neural Computation*, *13*(2), 357-387.

Zhang, H., Tanie, K., & Maekawa, H. (1996, April). Dextrous manipulation planning by grasp transformation. In *Proceedings of the Int. Conference on Robotics and Automation (ICRA)* (pp. 3055–3060). Minneapolis, USA.

Zöllner, R., Asfour, T., & Dillmann, R. (2004, September). Programming by Demonstration: Dual-Arm Manipulation Tasks for Humanoid Robots. In *Proceedings of the Int. Conference on Intelligent Robots and Systems (IROS)* (pp. 479–484). Sendai, Japan.

Zöllner, R., Rogalla, O., Dillmann, R., & Zöllner, J. (2001, September). Dynamic Grasp Recognition Within The Framework Of Programming By Demonstration. In *Int. Workshop on Robot and Human Interactive Communication (ROMAN)* (pp. 418–423). Bordeaux and Paris, France.