Detection and Recognition of Human Manipulation Building Blocks

Lisa Gutzeit

Dissertation

zur Erlangung des Grades eines Doktors der
Naturwissenschaften
- Dr. rer. nat.-

Vorgelegt im Fachbereich 3
(Mathematik & Informatik)
der Universität Bremen
im November 2022

Datum des Promotionskolloquiums: 08.03.2023

Gutachter: Prof. Frank Kirchner
Universität Bremen

Prof. Elisabeth André
Universität Ausburg

Abstract

New research challenges in robotics, which arise from attempts to get robots out of isolated environments into human spaces with a close human-robot interaction, demand new methods to acquire a deeper understanding of the human behavior and intentions. Especially in Learning from Demonstration (LfD), which provides an intuitive way to teach robotic systems new behavior from human movement examples, approaches are needed to recognize relevant movement segments in human motion data. In this way, building blocks of robotic motions can be learned which can be combined to generate a wide range of behaviors.

This thesis introduces algorithms to detect and annotate human building block movements in recordings of manipulation movements as well as an approach to determine hierarchical structures in these movements. The velocity-based Multiple Change-point Inference (vMCI) algorithm is presented, which identifies building blocks with a bell-shaped velocity profile using Bayesian Inference. The algorithm can be applied in an online and automatic fashion, without the need for expert knowledge, time-consuming training data generation or parameter tuning. To annotate the detected building blocks, several standard movement recognition approaches are compared with respect to their performance in the presence of only a small amount of labeled movement examples for training. It is shown that using 1-Nearest Neighbor (1-NN)-based movement classification, annotations can be found fast and reliably under these conditions. To detect basic movements as well as their concatenations into more complex, labeled actions, the velocity-based Hierarchical Movement Segmentation (vHMS) algorithm which hierarchically analyses human movements is presented.

Each of the developed methods is evaluated on different movement recordings acquired from several subjects, ranging from simple point-to-point movements to more complex dual arm object manipulation tasks. Furthermore, the application of the proposed algorithms in a framework to learn new robotic behavior from human

demonstration is shown. Different throwing motions are transferred to several robotic systems in a nearly automated way and in a reasonable amount of time. Additionally, the application of the segmentation approaches on teleoperated movements recorded with an exoskeleton is presented.

With the presented algorithms, the imitation of human behavior for robotic systems can be made more intuitive, automated, and more generally applicable. Furthermore, the hierarchical movement segmentation approach opens the door to construct hierarchical learning approaches which are based on human demonstration to learn complex robotic behavior more effectively.

Zusammenfassung

Neue Forschungsbereiche in der Robotik haben zum Ziel, Roboter raus aus isolierten Räumen in menschliche Umgebungen zu bringen, in denen sie eng mit dem Menschen interagieren. Dazu werden neuen Methoden gebraucht, um ein tieferes Verständnis über das Verhalten und die Intentionen des Menschen zu erlangen. Insbesondere in dem Bereich des robotischen Lernens aus Demonstrationen, bei dem Robotersysteme neue Verhaltensweisen anhand menschlicher Bewegungsbeispiele erlernen, werden Methoden benötigt, um grundlegende Bewegungsabschnitte in menschlichen Bewegungsdaten zu erkennen. So können Bausteine von robotischen Verhalten erlernt werden, welche zu einer Vielzahl von verschiedenen Bewegungen kombiniert werden können.

In dieser Arbeit werden Algorithmen zur Erkennung und Annotation menschlicher Bewegungsbausteine in Manipulationsbewegungen, sowie ein Ansatz zur Bestimmung hierarchischer Strukturen in diesen Bewegungen vorgestellt. Es wird der Algorithmus velocity-based Multiple Change-point Inference (vMCI) eingeführt, welcher Bewegungsbausteine mit einem glockenförmigen Geschwindigkeitsprofil mittels Bayes'scher Inferenz identifiziert. Der Algorithmus kann online und automatisiert angewendet werden, ohne dass Expertenwissen, zeitaufwändige Trainingsdatengenerierung oder Parameteroptimierung nötig ist. Zur Annotation der erkannten Bausteine werden verschiedene Standardbewegungserkennungsmethoden im Hinblick auf ihre Performanz unter Nutzung einer kleinen Menge an gelabelten Bewegungsbeispielen für das Training verglichen. Es wird gezeigt, dass mit 1-Nearest Neighbor (1-NN)-basierter Bewegungsklassifikation Annotationen unter diesen Voraussetzungen schnell und zuverlässig gefunden werden können. Um grundlegende Bewegungen, sowie deren Aneinanderreihung zu komplexeren, annotierten Bewegungen zu identifizieren, wird der velocity-based Hierarchical Movement Segmentation (vHMS)-Algorithmus vorgestellt, mit dem menschliche Bewegungen hierarchisch analysiert werden können.

Jede der entwickelten Methoden wird an verschiedenen Bewegungsaufzeichnungen von mehreren Probanden evaluiert, die von einfachen Punkt-zu-Punkt-Bewegungen bis hin zu komplexeren zweiarmigen Objektmanipulationsaufgaben reichen. Darüber hinaus wird die Anwendung der präsentierten Algorithmen in einem Framework zum Erlernen neues Roboterverhaltens aus menschlichen Demonstrationen gezeigt. Verschiedene Wurfbewegungen werden nahezu automatisiert und in angemessener Zeit auf verschiedene Robotersysteme übertragen. Zusätzlich werden die Segmentierungsansätze auf teleoperierte Bewegungen angewandt, welche mit einem Exoskelett aufgezeichnet wurden.

Mit den vorgestellten Algorithmen kann die Nachahmung menschlichen Verhaltens für Robotiksysteme intuitiver, automatisierter und genereller anwendbar gemacht werden. Darüber hinaus wird durch den Ansatz der hierarchischen Bewegungssegmentierung die Möglichkeit geschaffen, hierarchische Lernansätze zu konstruieren, welche aus menschlichen Bewegungsdemonstration komplexes Roboterverhalten effektiver erlernen.

Acknowledgments

During the journey towards this dissertation, many people supported me. I would like to thank all these people without whom this work would not have been possible.

First of all, I would like to thank my supervisor Prof. Frank Kirchner. He gave me the opportunity to work as a research assistant in robotics over all the years and to write this dissertation during that time. Additionally, I would like to thank Prof. Elisabeth André, who kindly agreed to be co-supervisor of this thesis.

Special thanks goes to all the colleagues who supported and encouraged me during my PhD years. This includes the other PhD-students, who gave me a lot of valuable feedback at the PhD-retreats as well as all the other colleagues and the many nice conversations and exchanges during the short and longer breaks in the daily scientific routine.

I thank Elsa Kirchner for all the exchange of ideas and her supervision in many projects I worked on. Especially working the project BesMan was an important stage in the creation of this thesis and I thank all teams members of the project. Especially the team which developed and implemented the BesMan learning platform together with me: Constantin Bergatt, Jan-Hendrik Metzen, Alexander Fabisch, Marc Otto, Manuel Meder and Christoph Petzold. With your engagement and contributions, is was possible to transfer human movement examples to different robotic system not only in simulation but also in real world experiments. I also thank my former student Sebastian Hellmann for the initial implementation of the trajectory labelling software and Hussam Al Turjman and Felix Busch for assisting me in the optimization of that tool. This toll really was a game changer in the creation of ground truth data for evaluation. Additionally, I thank all the persons who reviewed the final thesis, especially Su-Kyoung Kim and Teena Hassan for their detailed and valuable feedback.

Finally, I would like to thank my family and friends. I thank my mother for always supporting me in all my plans in life, so that I could always go the way I wanted to go. And I thank my husband, for whom it is simply a matter of course that both parents

take care of the everyday tasks with children, and my children who distracted me from time to time but mostly in a wonderful way. Without the support of my family and friends, this work would not have been possible.

Lisa Gutzeit,
Fall 2022

Contents

Part I.

Motivation and Goals

1 Chapter 1. Introduction

In this thesis, new approaches to analyze human movements are proposed and used to generate robotic behavior from human demonstration. By observing and analyzing the human behavior, insights are gained that can be used to realize an intuitive and adaptive interaction between humans and robotic systems and to generate more human-like robotic behavior. In this chapter, a more detailed motivation is given by shortly summarizing current research challenges in human-robot interaction, robot learning through interaction and human-like behavior learning in robotics. Afterwards, the goals of this thesis are presented and it is discussed how the presented work contributes to these research challenges in robotics.

1.1. Motivation and Research Challenges

While robotic systems have already been deployed in industrial settings in the last decades, current research and developments strike for systems which closely collaborate with humans. In these scenarios, the robotic systems need to operate outside shop floors and labs but in human environments where they are not isolated. For this, the systems need to continuously adapt to changing environments and may face tasks that were not pre-programmed during design. Furthermore, the systems need to closely collaborate with the human to provide the best possible support. For this collaboration, the robotic system must be equipped with new capabilities, as described in the following section. Furthermore, the system can directly learn during the interaction with a human, which is described in more detail afterwards.

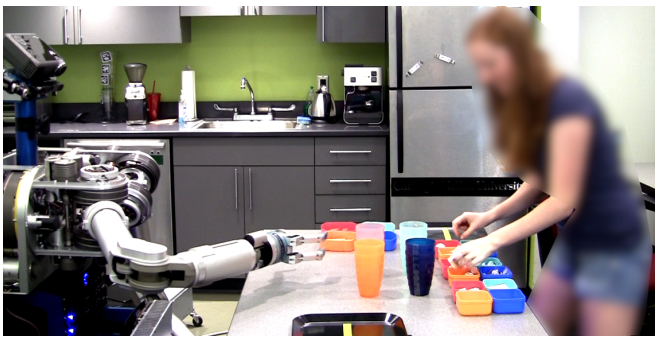


Figure 1.1.: Human-robot collaboration in a kitchen environment. *Image extracted from [Dragan et al., 2015], Figure 1.*

1.1.1. Towards an intuitive human-robot interaction

New and future applications demand a close collaboration of robotic systems with humans, e.g., in the support of elderly, as assistants at home (see Figure 1.1) or in co-work situations in industry [Schaal, 2007]. This leads to a closer interaction of the system with non-experts, which makes an intuitive control and interaction with the systems desirable. To enable a close interaction with humans, not only the robotic systems must be equipped with enlarged dexterities and control mechanisms that allow intuitive and safe interaction [De Santis et al., 2008], but also the human intentions, behaviors, and habits must be better understood [Kirchner et al., 2015]. Only if the human behavior can be understood in detail, the system will be able to react in an appropriate way and a natural and intuitive interaction can be realized. For this, the robot has to recognize its environment, which includes the recognition, tracking, and segmentation of objects [Yan et al., 2014] and additionally the human interaction partner and its behavior has to be recognized, understood, and predicted.

Human motion recognition is one of the most active research areas, with application in different fields ranging from gaming, animation, and sport analysis to surveillance applications. In robotics, human motion recognition not only plays an important role in human-robot collaboration but also in other human-robot interfaces. On different modalities, such as RGB or depth videos, marker-based motion tracking, and data gloves, a wide range of methods are applied to recognize the human and its motion. Surveys on different methods and applications are, e.g., given in [Poppe, 2010] or [Zhang et al., 2019]. However, most of these methods are supervised and

require a lot of training data.

In order to achieve a fluent human-robot collaboration, where the robot directly reacts to the human behavior, not only the current human action needs to be recognized, but also the human intentions that include goals, desires, and plans, need to be inferred from the observed human action [Wang et al., 2013]. An overview of recent approaches is given in [Kong and Fu, 2022].

Another important aspect of human-robot interaction is the generation of appropriate robot movements. In human-robot collaboration, the way the robot moves has a direct influence on the quality of the interaction [Dragan et al., 2015]. Many of these collaborations replace interaction tasks which were previously performed as human-human collaboration, e.g., in handing over tasks. As a result of long experience of these tasks, humans expect specific behavior and movements from their collaboration partner. If these are not matched by a robotic collaborator, the collaboration can be interrupted which could lead to safety issues. Studies showed that the collaboration of a human with a robotic system is more fluent if the human can predict robot motions [Dragan et al., 2015, Koppenborg et al., 2017]. Furthermore, movement speed has an influence on human-robot cooperation [Koppenborg et al., 2017]. Predictable motions can be generated using motion controllers, such as those presented in [Dragan et al., 2013], or by generating human-like robotic motions that humans are accustomed to through multiple interactions with other humans. Motion controllers which generate human-like movements are often designed for very specific tasks with limited opportunities for generalization whereas a machine learning based generation of human like-motions allows for some generalization. An overview of different approaches is given in [Gulletta et al., 2020]. For example, in [Gäbert et al., 2021], a motion planning approach is proposed which considers human joint limits and human arm movement characteristics to generate human-like arm movements while reaching the goal positions and avoiding obstacles.

1.1.2. Robot learning through interaction

One part of human behavior learning is learning through social interaction such as through imitation. This type of learning has gained a big interest in robotics, as it provides an intuitive mechanism to teach robots new skills. Learning through interaction is adressed, for example, in [Hörnstein et al., 2010], where language is

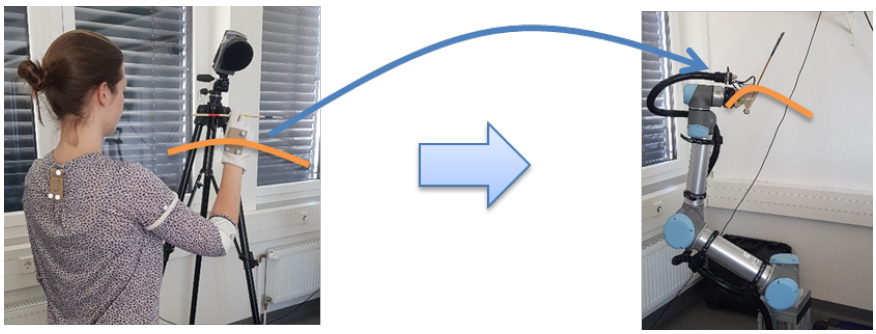


Figure 1.2.: Imitation of a stick-throwing movement. The movements of the human are recorded with marker-based motion tracking (left image) so that the movement trajectory (orange line) can be imitated by the robotic system (right image). Details are explained in chapter 7.

acquired for a humanoid robot during interaction with a human. Similar to human language acquisition in the interaction of adults with infants, an approach is developed which enables a robot equipped with embodied models of the infants' ears, eyes, vocal tract, and memory functions, to learn new words and their meanings though interaction with a human.

But also new movements can be acquired for robotic systems during interaction. In the area of robotic movement generation, several methods for Learning from Demonstration (LfD), also known as imitation learning, were published in the last years. In these methods, the human teaches a robotic system new behavior in an intuitive way that does not require expert knowledge. Motions to be learned can be taught directly by moving the robot extremities or indirectly by recording human movements to use them as movement examples from which new behavior is learned, as shown in Figure 1.2. An overview of several methods is given in [Argall et al., 2009]. Matarić proposed a biologically inspired model for imitation learning [Matarić, 2002]. This approach includes the perception of relevant movements by extracting salient features from visual information and a mapping of the visual input to an executable motor command. Additionally, a motor control system that consist of motor primitives which can be combined or superimposed to generate different movements is included as well as a movement classification mechanism which maps observed movement to executable movement to reuse already modeled motor programs. In their work, discrete, oscillatory as well as posture primitive should be modeled. They extract discrete as well as oscillatory movement from observed movement data

using clustering approaches. Parts of their model have already been implemented and evaluated, including an automated approach to derive motion primitives from observed behavior [Jenkins and Matarić, 2003].

More recently, a parametric representation of motion primitives called Dynamical Movement Primitives (DMPs) was introduced [Ijspeert et al., 2013] which encode arbitrarily shapeable, goal-directed trajectories. Different variants of DMPs were developed [Kober et al., 2010, Mülling et al., 2013], which can be adapted to explore different movements using reinforcement learning [Kober and Peters, 2012, Deisenroth et al., 2013]. Another way to face the tasks of learning of new robotic movements through imitation is the consideration of semantic information of behavior to be learned [Ramirez-Amaro et al., 2017]. This takes the importance of the relation between the movements and the involved objects into account which can, for instance, be important in household environments.

If the robotic movement should directly be learned from human movement observations, the correspondence problem must be solved, which is the problem of mapping human body positions to their robotic counterparts [Nehaniv and Dautenhahn, 2002]. Additionally, the relevant parts of the human movement demonstrations must be extracted from the movement observation. This can be done manually or by using automated approaches based on, for example, semantic information [Bouchard and Badler, 2007, Ramirez-Amaro et al., 2014] or probabilistic machine learning [Chiappa and Peters, 2010, Kulić et al., 2012, Fox et al., 2009, Lioutikov et al., 2017]. A more detailed overview will be given later in this thesis in section 4.1. Although, these methods segment human movements, it is in general not clear if the detected segments are the central movements of the demonstrated behavior.

1.1.3. Human-like learning of complex tasks

Artificial intelligence approaches achieved a great success in the last years and were able to solve tasks at a human-level performance [Lake et al., 2017]. This is mainly due to the use of deep neural networks, which obtained remarkable results in object or speech recognition [LeCun et al., 2015] or in playing of video games [Guo et al., 2014]. Also in robotics, there has been an impressive progress, for example, in learning in-hand manipulation of a cube [Andrychowicz et al., 2020]. However, in several areas robotics systems are still far away from learning and behaving like a human. Solving

a task like grasping a cup can easily be performed by humans independent of, e.g., the orientation of the cup. For a robot, even minor changes to the task or the objects involved could require the use of new, accordingly adapted control mechanisms. With increasing complexity of the task and the system, the control of robot movements gets even more challenging.

Next to learning through interaction, also other principles of human learning can be used to learn new robotic behavior. Learning of complex behaviors in humans takes place incrementally, as shown in behavioral studies [Sakai et al., 2003]. This means that smaller individual building blocks are learned separately and later combined to a single, higher level complex behavior. This process is called chunking of action repertoires [Graybiel, 1998]. Hierarchical reinforcement learning is an approach to address the learning of complex tasks for artificial systems by decomposing learning problems into subtasks. A recent overview of current approaches is, for example, given in [Pateria et al., 2021]. In contrast to this, most state-of-the-art LfD methods are monolithic learning approaches, i.e., they learn one behavior that covers the whole demonstration. In order that hierarchical approaches can be used in LfD in the future to transfer the principle of hierarchical learning of complex tasks using simple building blocks to a robotic system, the subtasks of a demonstrated human behavior need to be identified. This is the main goal of this thesis, as detailed in section 1.3.

1.2. Terminology

To better understand the goals and contributions, three important terms used during this thesis are defined as follows:

Definition: Manipulation Movement

Hand or arm movement executed to manipulate an object. In this thesis, it is focused on these discrete, goal-directed movements. Examples for manipulation movements are pick-and-place tasks or dual arm manipulations such as pouring water into a cup. Rhythmic arm movements, which would be executed, e.g., during a stirring movement as well as whole body motions such as walking or dancing are not considered in this work.

Definition: Building Block

A movement building block is a central movement entity of manipulation which can be combined with other building blocks to solve different tasks. Building blocks can be seen as submovements during manipulation, such as reaching or placing. The observable characteristic of these building blocks is a bell-shaped hand velocity, which will be shown later in this thesis.

Definition: (Movement) Action

Concatenations of multiple building blocks are named movement action within this thesis. Actions can be for example manipulation movements such as pour water into cup or cutting vegetable.

1.3. Thesis Goals

One central aspect for human-robot interaction and especially imitation learning is the detection of movement building blocks, which are also referred to as motor or motion primitives in the literature. If they are recognized in human motion, the current behavior can be determined and the movement that may occur next can be inferred [Matarić, 2002]. This is important to achieve a precise reaction of a robotic system in human-robot collaboration tasks as described in section 1.1.1. Additionally, building blocks are good models for imitation learning problems as described in section 1.1.2, because they can be combined to different movements.

In this thesis, human manipulation movements are automatically analyzed to detect movement building blocks and their connections. For this, two central questions are addressed: (a) What are these building blocks? and (b) How can they automatically be detected in observed human behavior despite the huge variances in different executions of the same task. To achieve a board applicability, approaches should be developed which are applicable in different situations with low user input and minimal requirement of manual optimization. The detected building blocks should be transferred to a robotic system to generate central elements of robotic behavior from human demonstration.

The **goal** of this thesis can thus be formulated as follows:

Development of methods to detect building blocks in human manipulation movements that can be used to generate robotic behavior.

To achieve this goal, several subgoals must be fulfilled and are addressed within this thesis:

Subgoal (1) *Unsupervised segmentation of human manipulation movements to detect building blocks.* Development of an approach to segment recorded movement trajectories into building blocks. To reduce manual user input, the approach should work unsupervised and only with a small number of (hyper-) parameters that need to be tuned. Part of the development of this algorithm is the identification of useful features in movement trajectories of manipulation building blocks.

Subgoal (2) *Few-shot recognition of building blocks.* Development or identification of methods to classify detected building blocks to indicate what kind of movement is currently observed. To minimize the effort needed in the generation of training data, recognition methods should be used which classify movement segments using a small number of labeled training data.

Subgoal (3) *Recognition of connections between building blocks.* Development of an approach to hierarchically segment manipulation movements. In a hierarchical segmentation, the building blocks of the demonstrated movement should be detected as well as their combinations to more complex actions to generate a deeper understanding of the composition of the human behavior.

Subgoal (4) *Generation of robotic behavior based on human movement building blocks.* Integration of the developed methods into a framework to generate robotic behavior from human movement demonstrations. With this, the application of the developed segmentation and recognition approaches should be shown.

1.4. Contributions

With the detection of building blocks in human movements and the connection between these, this thesis contributes to all three research areas described in section 1.1. A more detailed knowledge about the current human behavior can be achieved by detecting the currently performed building block and the action this building block belongs to. This can be used to develop a more intuitive human-robot interaction and to generate robotic movements from human examples in an efficient way.

The velocity-based Multiple Change-point Inference (vMCI) algorithm is developed in this thesis to automatically detect movement building blocks in human manipulation movements. An open source implementation of the algorithm is available at GitHub. The algorithm works unsupervised, without the need for manual adaption of, e.g., hyper-parameters, also on movement recordings obtained from different modalities or different movement tasks. That means, no time intensive manual training data generation is needed and the algorithm can be run on different manipulation movements or different datasets without manual hyper-parameter adaption. Using this algorithm, the building blocks of movements demonstrated by the human can be detected, which can be directly used to generate robotic behavior. The idea is that by detecting the building blocks needed to solve different manipulation tasks, a robotic system can be equipped with these building blocks and is then able to solve different tasks by combining them in different ways. This thesis contributes to this with the first step needed: the detection of building blocks in human movement demonstrations.

The vMCI algorithm segments human motion data into building blocks but without relating these building blocks to certain movements, e.g., by giving them movement labels. This next step is called *movement recognition* within this thesis. It is needed to understand what the human is currently doing. Additionally, movement segments that should be used for imitation in LfD applications can intuitively be selected based on assigned movement annotations. To recognize human behavior in video or image data, many approaches were proposed in the last decades [Poppe, 2010]. Most of these approaches benefit from a huge amount of available data. In contrast to human activity recognition in the wild, smaller movement entities, such as a specific direction of approaching an object which should be grasped, need to be detected in LfD applications. To collect training data for these applications, movement demonstrations need to be recorded, preprocessed, and manually labeled. These

efforts can be minimized if so-called few-shot classification methods are used for movement recognition which can recognize various behaviors with a small number of training examples. Using these methods, training time can be minimized. This makes these algorithms additionally interesting for adaptive approaches in which newly observed movements are addressed using re-training. In this thesis contributes to this area by comparing different state-of-the-art movement recognition approaches with respect to their classification accuracies with a small number of training examples. For this, k-Nearest Neighbor (k-NN), Hidden Markov Model (HMM), and Long-short Term Memory-Network (LSTM)-based classification are compared on different datasets, some consisting of building blocks detected using vMCI and one dataset containing more complex gestures.

Additionally, this thesis contributes to the identification of the connections between detected building blocks. If not only the building blocks in observed movement recordings can be detected, but also how they are connected, i.e., which sequence of building blocks can be observed in different actions, a deeper understanding of the observed human behavior can be obtained. This could be used to generate robotic behavior similarly to human movement generation based on the combination of building blocks to perform different actions as described in section 1.1.3. For this, the hierarchical segmentation approach velocity-based Hierarchical Movement Segmentation (vHMS) is developed, which identifies building blocks as well as their combinations to more complex actions in observed human manipulation movements. This algorithm forms the basis for the generation of hierarchical learning approaches for robotic movement generation based on human examples.

The contributions of this thesis are not limited to the analysis and automatic segmentation and recognition of human manipulation movements. Additionally, the developed methods are integrated into a framework for LfD, which can be used to directly transfer the detected building blocks to a robotic system. It will be shown that useful and successful robotic movements can be learned from the detected human movement building blocks. Additionally, the application of the developed approaches on movements obtained using an exoskeleton during teleoperation is shown. This opens the door for future applications which use the automatically determined movement segments, e.g., to learn robotic behavior directly during teleoperation of the system.

An overview of all contributions and their grading from human movement anal-

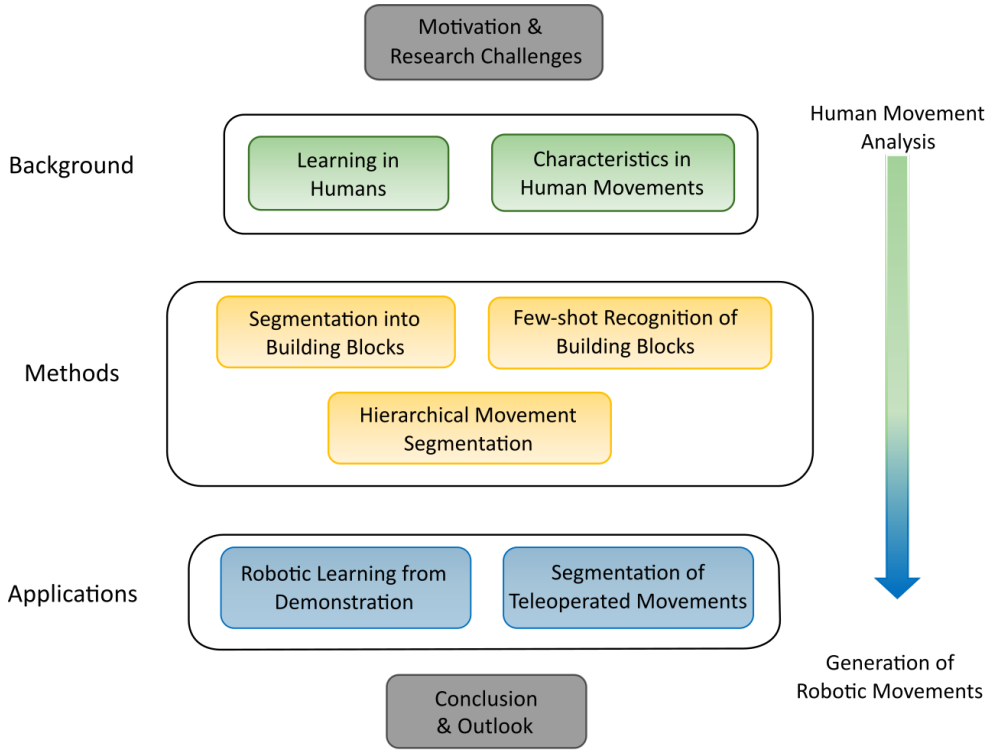


Figure 1.3.: Overview of the structure and contributions of this thesis. After starting with the motivation and research challenges, the background is presented, which includes a summary of neurobiological studies and experiments to characterize movement building blocks. The gained insights are then used to develop automated analysis methods for segmentation and recognition of human manipulation movements. Afterwards, the developed methods are applied and integrated to generate robotic movements from human examples. At the end, the thesis is completed with the conclusion and an outlook.

ysis to the generation of robotic movements is shown in Figure 1.3. To validate the developed methods and to compare them to other state-of-the-art methods from the literature, several human manipulation movements were recorded using marker-based motion capturing. The recorded movements range from simple point-to-point movements to more complex dual arm manipulation tasks. For evaluation of the approaches, manually determined segment borders and movement labels are required. To generate these, a software tool for trajectory visualization and simplified trajectory labeling is developed.

Most of the work presented in this thesis was already published in several publications. In each chapter, it is stated in the introduction which publications are related to the presented work. In appendix B, all publications are listed and details about my contribution to each of these are given.

1.5. Thesis Structure

This thesis is divided into four parts. **Part I** is comprised of the previous sections which motivate the thesis and describe the goals and contributions. Additionally, the neurobiological background is presented in chapter 2, which gives details about human movement generation, the way humans learn new behavior and the main characteristics of human manipulation movements. At the end of that chapter, in section 2.4, it is discussed why the bell-shaped velocity of the hand is selected as the main feature to characterize manipulation building blocks. In the overview diagram of Figure 1.3, this Part I is colored in green.

Part II introduces the developed methods for automated analysis of human motions, marked in yellow in Figure 1.3. In chapter 3, the marker-based movement recording systems are described and the datasets are introduced that were acquired for the evaluations of the developed methods. In section 3.8, the methods to generate a labeled ground truth for these datases to be used as reference for the automated segmentation and recognition approaches are described. Chapter 4 starts with an analysis of different features of human manipulation movements, before the vMCI algorithm is introduced which segments human movements into building blocks with a bell-shaped velocity profile. The algorithm is evaluated on several of the recorded one-arm manipulation movements. In chapter 5, different machine learning methods to classify human manipulation movements into known movement classes

are compared. Three algorithms, k-NN, HMM, and LSTMs, are evaluated with respect to their ability to classify building blocks obtained using vMCI with a small number of labeled examples used for training. In chapter 6, vHMS is introduced which is developed to infer the connections between building blocks and their combination to movement actions. The approach is evaluated on simple one-handed movements as well as on more complex dual arm motions in which building block movements of each hand are combined to perform a bimanual rotation task.

In **Part III**, two applications of the developed segmentation and recognition approaches are presented, which are colored in blue in Figure 1.3. In chapter 7, a learning platform to learn robotic behavior from human demonstrations is introduced. The vMCI segmentation as well as the movement classification approaches are integrated into this framework to detect the movement sequences in human demonstrations which should be transferred to a robotic system using imitation and reinforcement learning. The learning platform is evaluated on two different types of throwing movements, which are transferred to different robotic systems. In chapter 8, vMCI and vHMS are run on movements obtained during teleoperation of a humanoid robotic system. In this application, the developed algorithms are tested on challenging movement data, which are not perceived directly from human body positions but indirectly through an exoskeleton. Thus, the human is restricted in the execution of the movements which causes more irregularities and variance in the recorded movement trajectories.

In **part IV**, a summary as well as a conclusion are given in chapter 9. Future work is discussed in the outlook in chapter 10. The last part is the appendix, which includes a list of publications published during the preparation of this thesis.

2 Chapter 2. Neurobiological Background

In this chapter, the basic principles and most established theories about the computational mechanisms of human movement generation and adaptation through learning are explained. A short literature review is given to highlight the different theories and the complexity of the generation of goal-directed human motions. This gives a basic understanding of how observable human behavior, which can for example be recorded via motion tracking, may be generated by the human central nervous system (CNS). In the presented overview, the focus is on the generation and adaptation of goal-directed reaching movements, to which the approaches in this thesis refer.

Next to the short review of the computational principles of the generation of goal-directed human movements, the literature dealing with the investigation of characteristic patterns in these movements is presented. This is a fundamental background of the human movement analysis part of this thesis, as these characteristics are used to motivate the algorithms introduced in the later chapters. At the end of this chapter, a short summary is given and the presented theories and literature is set into relation to the presented thesis. A more detailed overview of the computational principles of human movement generation is given, for instance, in [Wolpert and Ghahramani, 2000] or [Shadmer and Wise, 2005b].

2.1. Human Movement Generation

To generate a specific movement needed to achieve a certain task, the CNS not only has to generate the appropriate motor commands but it also needs to measure the

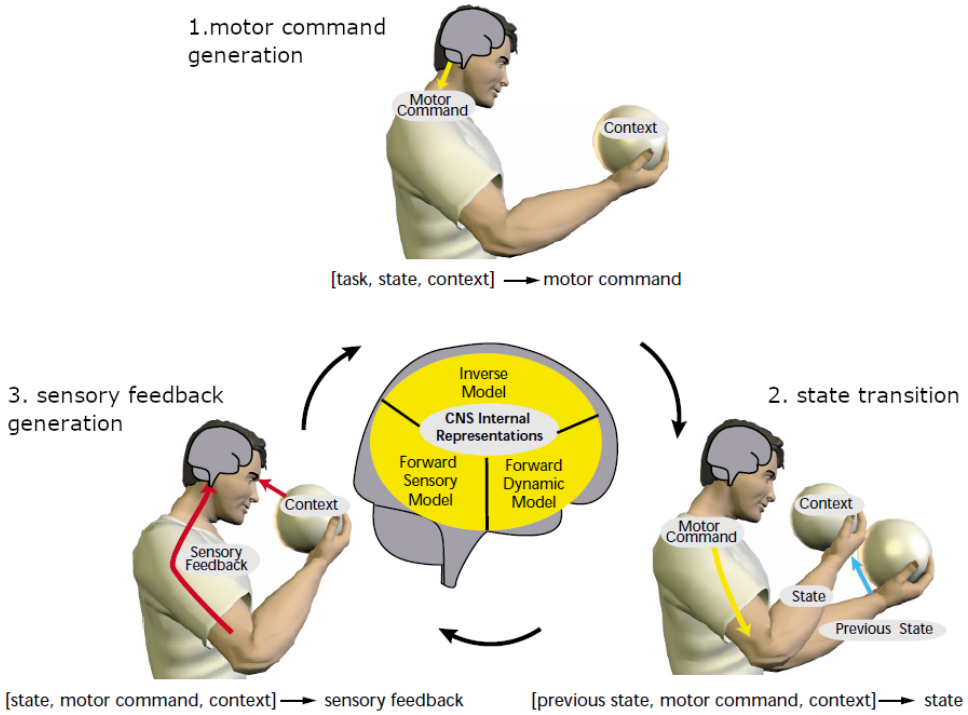


Figure 2.1.: Sensorimotor loop and involved internal models in the CNS. 1: Based on tasks and environmental states the CNS generates motor commands using inverse models. 2: The resulting next state of the arm can be predicted by the CNS using forward dynamic models. 3: The execution of the motor commands causes a change in the position of the arm and the involved object, which can be measured suing sensory feedback and the forwards sensory model of the CNS. *Image adapted from: [Wolpert and Ghahramani, 2000] Figure 1.*

consequences during movement execution. It is assumed that the CNS generates internal models for this, which are included in the sensorimotor loop [Wolpert and Ghahramani, 2000]. The sensorimotor loop models the relation between motor commands and sensory consequences as shown in Figure 2.1. It consists of three phases. In the first phase, named *command generation*, the motor commands are generated based on the given task and the current state of the system and the environment. Secondly, in the *state transition*, the execution of the motor commands results in a change of the state of the system and the environment. In the last phase, called *sensory feedback generation*, these new states are predicted or measured using the sensory system of the human. It is assumed that each of these phases are represented as internal models in

the CNS.

In the example task of grasping an object, the processing steps needed to be performed by the CNS in the sensorimotor loop can be described as follows: (1) The movement of the hand towards target object is planned and the corresponding motor commands are generated. (2) Based on the motor commands, the reaching movement is executed and the arm starts to move. This induces a change of the state of the arm and later during grasping also of the object which should be manipulated. (3) These state changes are measured or predicted by the CNS to ensure that the task of grasping the object is performed correctly. (4) In the case of errors, i.e., incoherences between excepted and measured states, new motor commands are generated to correct the movement by restarting the sensorimotor loop. In the following sections, the theories and studies about the realization of each step (movement planning, generation of motor commands, state estimation, and prediction) are explained.

2.1.1. Movement planning

There are infinitely many possible ways to move the arm and hand. This is also the case for goal-directed movements such as grasping an object. Despite the huge possibilities to vary the movement trajectories, the CNS always generates very similar movements. It is assumed that this is because the CNS generates movement with low cost, a theory which is called optimal control [Diedrichsen et al., 2010]. However, it is an open question what optimization function the CNS uses to generate movements with low cost. Flash and Hogan proposed in the 80', that arm movements are generated by maximizing smoothness in the hand trajectory [Flash and Hogan, 1985]. In their experiments, empirical data consisting of straight as well as curved arm motions could be reproduced by minimizing the jerk of the trajectories, which is the second derivative of the movement position. However, it remained unclear how the CNS measures the smoothness for optimization [Wolpert and Ghahramani, 2000]. Later, another cost function for goal-directed eye and arm movements were proposed, which is based on minimizing the movement error [Harris and Wolpert, 1998]. The idea in this work is that the observed direct and smooth movements are generated using short movement commands, which reduces noise in the movement execution. By reducing noise, movement errors can be minimized. In this model the smooth trajectories are an observable side-effect.

2.1.2. Generation of motor commands

There are several hypotheses on how the motor commands are generated by the CNS in order that a desired movement is performed. One of the first was the equilibrium point hypothesis [Flash, 1987]. An equilibrium point is defined as the situation in which opposing muscles are in a state of balance with each other. In the equilibrium point hypothesis, it is assumed that movements are generated by the CNS as successive equilibrium points of the limp on a trajectory. However, this hypothesis has been controversially discussed in the literature, since a high stiffness of the human limbs would be needed to effectively generate arm movements based on this idea [Wolpert and Ghahramani, 2000]. Instead, the equilibrium points are assumed to be a stabilizing mechanism in [Shadmer and Wise, 2005a].

Another popular theory suggests that the motor commands are generated using inverse models which map the desired state at every point on the trajectory to the corresponding motor commands [Kawato et al., 1987, Kawato, 1999]. Using inverse models, only little stiffness of the system is needed [Wolpert and Ghahramani, 2000]. Due to a study indicating that the human limps have low stiffness, this theory is preferred by some experts compared to the equilibrium point hypotheses, because in the latter complex trajectories would be required for simple movements [Gomi and Kawato, 1996, Wolpert and Ghahramani, 2000].

In addition to these two opposing theories, there is the assumption that movements are generated based on a small number of motor primitives in the spinal cord. The theory arises from experiments performed on frogs [Bizzi et al., 1991, Giszter et al., 1993]. In these experiments, frogs were spinalized to directly perform an electrical stimulation in the lumber spinal cord of the frog which provokes a movement in the hindlimb. Using a six-axes force transducer placed on the ankle of the frog, the resulting forces were measured. The limp was placed at different locations within the legs workspace to record the direction and amplitude of the force generated at these positions, see Figure 2.2(a). From this, a vector field of generated forces for the workspace of the frog's leg was determined. During the experiments a small number of convergent force fields were detected. Two of these force fields can be seen in Figure 2.2(b). In [Mussa-Ivaldi et al., 1994], the resulting vector field of the simultaneous stimulation of two areas in the spinal cord was compared to the mathematical summation of the vector fields resulting from the individual stimulation

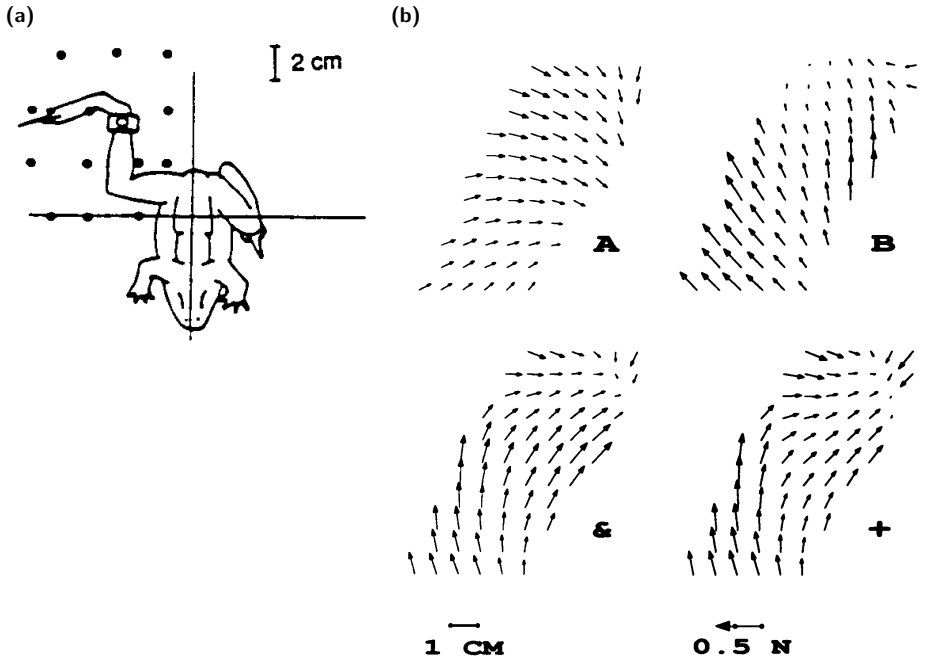


Figure 2.2.: Force fields generated in splinalized frogs which lead to the idea of movement generation based on a small number of motor primitives. (a) Positions where the limp of the frog was placed before stimulation of a certain area in the spinal cord. At the ankle of the frog, a transducer measures the resulting forces. (b) Resulting vector fields. *Field A* and *Field B* were generated by stimulating two different positions in the spinal cord. Field & is the result of the simultaneous stimulation of these two positions. Vector *Field* + is the mathematical summation of *Field A* and *Field B* and looks very similar to *Field &*. *Image sources: (a) [Bizzi et al., 1991] Figure 1A, (b) [Mussa-Ivaldi et al., 1994] Figure 3 (left).*

of the same areas. The results can be seen in Figure 2.2(b). Interestingly both, the simultaneous stimulation as well as the mathematical summation, result in a similar vector field. From this, the hypothesis was put forward that movements are generated from a small number of motor primitives, which is supported by several other similar experiments [Mussa-Ivaldi and Bizzi, 2000]. Each primitive is in this hypothesis represented by a certain area in the spinal cord which generates a convergent force field. By activating several of these areas in combination, a variety of different movements are generated.

2.1.3. State estimation and prediction

The last part needed in the sensorimotor loop is the estimation of the current state of the human and the manipulated environment in the CNS. This can be measured using sensory signals, such as the visible information captured by the eyes. However, sensory signals reach the CNS with delays and can be noisy or provide only partial information. An alternative is to predict the current state of the limps using a forward model, which maps the performed movements to its consequences using a copy of the motor commands and a model of the involved dynamics. Empirical studies, for example two studies examining human hand motions [Wolpert et al., 1995b, Beers et al., 1999], suggest that these two approaches, the forward model and the feedback generated by the sensory inputs, are combined into a so-called observer framework to estimate the current states. An example of this observer framework is the Kalman filter, in which the current state is determined from the estimation of the previous state and a model of the motor command and the involved dynamics. Based on this prediction of the current state, the expected sensory feedback can be determined. In case of differences to the actual sensory feedback, the current state estimate can be corrected.

2.2. Learning in Humans

Humans can adapt their behavior to different situations and to learn motions which can effectively be generated in recurring situations. A good overview of different learning and adaption mechanisms is given in [Shadmer and Wise, 2005b], especially in chapter 4 of this book. The authors combine the different learning mechanisms under the term *motor learning*, which includes the adaption of motor commands to changing environments, as well as mechanisms to acquire skills, learning of instinctive behaviors, and decision making. The first of these terms, motor adaption, comprises the changes in the existing motor commands triggered due to changing context related to the tasks. If the existing motor commands, referred to as motor repertoire in [Shadmer and Wise, 2005b], is extended this is called skill acquisition or skill learning. This happens in each individual during its lifetime as well as over generations.

If it is looked at motor adaption from the perspective described in section 2.1, in which movement generation takes place in the context of the sensorimotor loop using

internal models, motor adaption refers to the adaption of the internal models to new situations. For example, an internal model which generates the movement to grasp a ball needs to be adapted if the context changes, for example the position of the ball, its size or if another object should be grasped. Additionally, changes of the system must be considered, which origin for example of changes in the size of the limps due to growth. To learn motor commands for these new properties in the sensorimotor loop, the sensory input can be used to generate a feedback learning model [Wolpert and Ghahramani, 2000]. In this model, the error between desired and estimated states are used to update the inverse model to the new context. Using this feedback-error learning, internal models can be adapted to a changing environment. Wolpert et al. suggest that internal models for new contexts are generated using combinations of existing internal models generated or adapted for specific situations or contexts. In this understanding, multiple internal models are treated as building blocks to generate behavior for different situations. Experiments examining the learning of reaching movements in environments with modified kinematic and dynamic properties support the theory of combination of internal models during learning [Flanagan et al., 1999].

The adaption of behavior is studied in more detail in motor sequence learning tasks. Graybiel et al. investigated mechanisms in the basal ganglia, which contribute to behavior learning [Graybiel, 1998]. She suggests that during learning of new movements, known motor actions represented in the striatum of the brain are chunked together to learn movements for new situations. From these processes in the brain, it is deduced that complex human behavior is generated from known blocks, a mechanism called chunking of action repertoires. In the selection of chunked actions, the basal ganglia may play an important role. This is supported by several behavioral studies. For example, in [Adi-Japha et al., 2008] learning experiments were conducted, in which the participants had to learn a certain movement sequence in which the tip of the thump had to be touched with each other finger in a predefined order. The results do not support the proposed hypothesis that the movement sequence is learned withing a single learning process. This suggest that the movements are learned incrementally by concatenating consecutive movement elements referred to as chunks. This is also supported by further behavioral studies, e.g., the results in [Sakai et al., 2003] indicate, that chunking is important for the representation of long motor sequences in the brain.

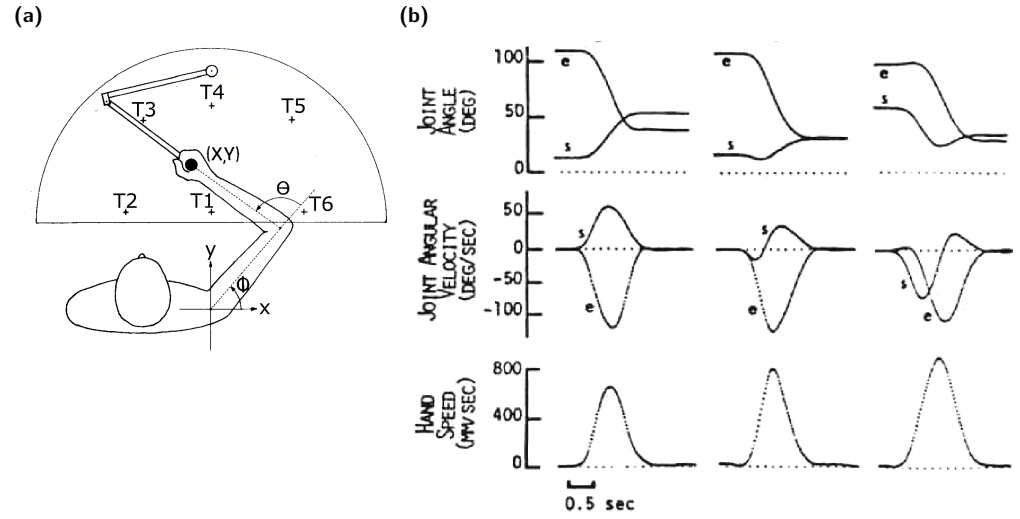


Figure 2.3.: (a) Experimental setup to determine the characteristics of point-to-point movements performed in [Morasso, 1981]. (b) Resulting joint angle positions for the elbow (e) and shoulder (s) joint angular velocities and absolute hand velocity for three different point-to-point movements. First column: movement from T1 to T4; Second column: movement from T1 to T5; Third column: movement from T2 to T5. Whereas the joint angular velocities show different patterns with respect to the position of the points, the absolute hand velocity is always bell-shaped. *Image sources: (a) adapted form [Morasso, 1981] Figure 1, (b) [Mussa-Ivaldi and Solla, 2004] Figure 1 (A).*

2.3. Characteristics of Human Movements

In the previous section, different hypotheses from the literature were introduced, indicating that human movements are generated from a small number of motion primitives, which are activated with respect to internal models to fulfill a certain movement task. However, the full cycle of movement generation is an ongoing research topic and not every step is or can be proved. If the human movement resulting from the neurobiological mechanisms should be investigated, the analysis of external observations of the performed movement suggests itself because these observations can easily be acquired, e.g., by using videos of the movements or motion tracking. As described in section 2.1.1, the CNS plans movements in a way, that the hand follows a nearly straight path with smooth trajectories. As observed in several experiments, these trajectories show a bell-shaped velocity profile. In this section,

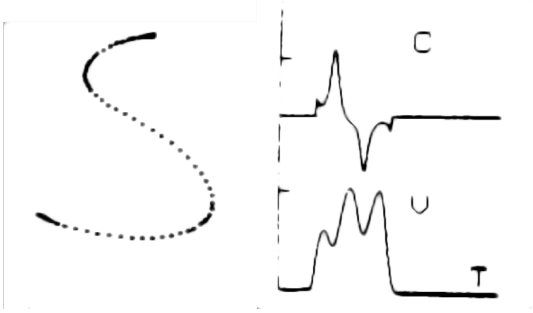


Figure 2.4.: Drawn trajectory with inflection points (left) and resulting curvature (C) and velocity (V) profile. For this curved trajectory, multiple velocity peaks can be observed. *Image source: [Morasso and Mussa-Ivaldi, 1982] Figure 7 (middle).*

the main experiments conducted in the literature in analyzing common observable characteristics of human goal-directed arm movements are explained.

Studies performed by Wolpert et al. showed that the CNS generates movements which lie on a path with slight curvature in Cartesian coordinates but on a straight path in visual coordinates [Wolpert et al., 1994, Wolpert et al., 1995a]. Possibly, this results from a distortion of the visual system in the perception of the Cartesian space [Shadmer and Wise, 2005b]. These nearly straight movements of the hand show a recurring pattern, which was firstly reported in [Morasso, 1981]. In their work, the characteristics of point-to-point movements were analyzed using potientometers of two shafts representing the lower and upper arm movement, as shown in Figure 2.3 (a). The results for three different point-to-point movements can be seen in Figure 2.3 (b): While the recorded joint angle positions and velocities for the elbow and shoulder joint are different for different movements, the position of the hand changed monotonically with a bell-shaped velocity pattern. From this it is derived that the central commands generated by the CNS are in hand coordinates [Mussa-Ivaldi and Solla, 2004].

Further analyses on regularities of hand movements were conducted which are not limited to point-to-point movements. An early analysis on different handwriting patterns, including straight movements as well as curved or circular movements, is reported in [Morasso and Mussa-Ivaldi, 1982]. While also a symmetric bell-shaped velocity profile for point-to-point movements were observed, velocity profiles with multiple velocity peaks could be observed for curved movements, as shown in Figure 2.4. These multiple velocity peaks in handwriting and drawing movements show

a specific relation between the curvature of the trajectory and the velocity: the angular velocity of the movement is equal to the curvature to the power of two-thirds times a constant called the gain factor [Lacquaniti et al., 1983, Viviani and Cenzato, 1985]. In this regularity, called the two-thirds power law, the gain factor describes the average velocity of the movement and is constant between points of inflection. From these piecewise regularities it is derived that the observed hand movements during writing are compositions of submovements [Flash and Hochner, 2005], which are also described as overlapping segments with a bell-shaped velocity [Morasso and Mussa-Ivaldi, 1982]. All these observations suggest that the hand velocity is an important feature in hand movements and that observable bell-shaped velocity profiles may be a characteristic of submovements.

Several behavioral studies indicate that during learning of hand movements the number of submovements decreases and smoother trajectories with less velocity peaks are executed. This pattern could be observed in studies with stroke patients [Rohrer et al., 2004], babies performing reaching movements [Berthier et al., 2005] as well as in a study with healthy participants learning a sequence of point-to-point movements [Sosnik et al., 2004]. One of the resulting velocity curves during learning in the last study investigating the co-articulation of movement primitives in healthy adults can be seen in Figure 2.5. The image shows, that during learning the task of drawing a line through four different via points, the velocity curves change from single bell-shaped velocity curves for each of the four point-to-point movements to a velocity curve with only two peaks at the fifth day of learning. The straight path through the 4 points vanished and were replaced by two curved paths, resulting in less velocity minima and a reduced duration of the whole movement.

The relation between these observable regularities in movement execution and the generation of hand movements is not fully understood. It was assumed that the observable submovements originate in a segmented control of hand movements by the CNS [Viviani and Cenzato, 1985]. For rhythmic movements, which are for example performed during cyclic drawing movements, the segmented control hypothesis was challenged by [Sternad and Schaal, 1999]. In their experiments, human cyclic drawing movement could be reconstructed using a robotic arm with a continuous control mechanism. The conclusion for discrete movements is an open question. Richardson and Flash also could not verify the theory of segmented control [Richardson and Flash, 2002]. In their experiments, several arm movements were reconstructed math-

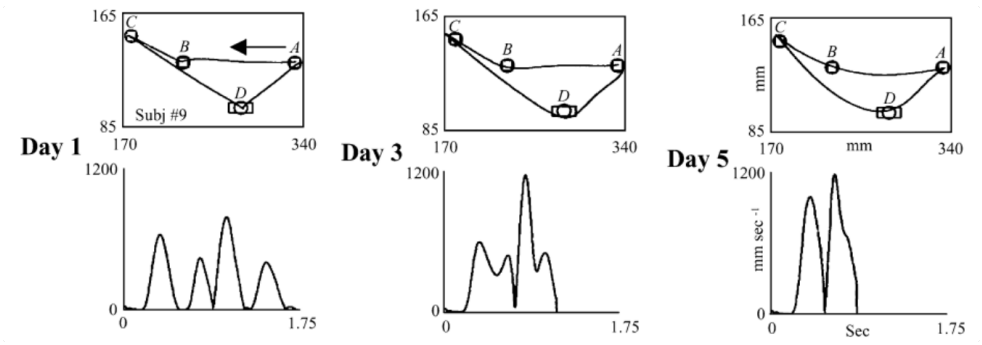


Figure 2.5.: Representative movement example of learning to connect four different points (top row) and corresponding velocity profiles (bottom row). During learning, the movement gets smoother with less velocity peaks. *Image source: [Sosnik et al., 2004] Figure 3A.*

ematically by minimizing several cost functions. Point-to-point as well as curved movements could be modeled using a single cost function, which contradicts the assumption of segmented control. This leads us back to the theory described in section 2.1.1, that the observed smooth trajectories are a result of a trajectory optimization mechanism performed by the CNS.

2.4. Discussion: How neurobiological theories may help to investigate human behavior for robotic applications

The previous sections described the complex computational mechanisms involved in human movement generation which are not fully understood. To generate goal-directed arm movements, the CNS needs to determine internal body states and it needs to estimate and process the sensory information about the environment to plan the movements and to monitor their execution. At the same time, the performed movements need to be adaptable to a continuously changing environment.

The reviewed literature in the previous sections show that there are strong indications that the movement generation mechanisms are accomplished using movement building blocks on different levels. On the neural level, the studies in spinalized frogs suggest that there is a small number of areas in the spinal cord which constitute building blocks for movement generation in the form of spinal force fields. These can be combined to execute a wide range of movements. On the level of movement

execution, observable regularities can also be interpreted as indicator for movement building blocks which supports the theory of generation of movement based on a combination of a small number of movement building blocks. This is in line with studies dealing with the generation of new movement sequences. As described in section 2.2, these studies indicate that humans do not learn complex movements as a whole, but successively by combining already learned movement blocks.

The internal mechanisms that underlay the movement generation based on primitives are not directly observable from outside. However, in order that a similar principle based on combination of building blocks can be used for robotic applications, the building blocks of human movements need to be identified. To this end, human movement demonstrations can be used as starting point to generate basic robotic movements, which can successively be combined to generate different robotic behavior which is similar to the human movement example.

To detect building blocks in human movements, observable regularities in the hand velocity can be used. These observable direct and smooth trajectories cannot be directly mapped to internal movement building blocks on a neural level but are an indication that the CNS plans movements based on an internal optimization principle. From this origins the reoccurring pattern in human manipulation movements, which is the bell-shaped velocity profile. This velocity pattern of the hand will be used in this thesis as main feature to separate human movement into individual entities. A bell-shaped velocity curve is observable in point-to-point movements but also in other arm movement as a superposition of single bell-shaped curves which result in multiple velocity peaks. The minimization of velocity peaks during learning could be seen as indicator, that already learned and optimized movement parts have one, or at least a very small number of velocity peaks. In chapter 4, this characteristic will be used to automatically detect movement building blocks in human manipulation movements. However, it is important to note that there is no prove that the observable movement parts with a bell-shaped velocity are referring to movement primitives. Rather, they are a result of the movement optimization performed by the CNS. Nonetheless, strong occurrence in point-to-point movements as well as their superposition when learning curved movements allow a reference to primitives. In this thesis, it is assumed that the analyzed manipulation movements are concatenations of movement entities with a bell-shaped velocity. This bell-shaped profile is not limited to movements with velocities starting or ending at time points with no movement where the velocity is

zero, as shown in Figure 2.3(a) and in the left velocity plot of Figure 2.5. In fact, an algorithm for detecting building blocks which are characterized by bell-shaped velocity curves with varying start and end velocities and symmetric as well as asymmetric profiles will be presented. With this, the movement characteristics of point-to-point movements as well as more complex movements with superimposed bell-shaped velocity curves are considered. It will be shown during this thesis, that the bell-shaped velocity is a promising feature to detect individual movement units were defined as movement building blocks in section 1.2. In chapter 7, it will be shown that these detected building blocks are suitable to be a template for robotic movement generation.

Part II.

Analysis of Human Manipulation Movements

3 Chapter 3. Acquired Datasets

For this work, several datasets of human manipulation movements were recorded and will be used in the upcoming chapters to evaluate the developed segmentation and recognition approaches. The movement data was acquired with the Qualisys motion capture system, which records positions of markers attached to the subjects at high precision, see section 3.1.1. One dataset consisting of several gestures was recorded using the inertial measurement unit (IMU) based Xsens motion capture suit described in section 3.1.2. Although the recorded gestures are not manipulation movements as defined in section 1.2, this dataset will be used to evaluate different methods for movement recognition with small training set sizes in chapter 5 because it comprises more complex movements with high inter-subject variations.

In total, 8 different motion datasets were recorded, ranging from simple movements in a restricted environment (step data, section 3.2), to complex dual arm manipulation tasks (object rotation data, see section 3.7). An overview of all recorded datasets is given in Table 3.1. For each dataset, the movements of several subjects were recorded, with multiple repetitions of the same movement. To generate labeled ground truth segments, some datasets were manually segmented using a visualization tool described in section 3.8. These datasets will be used in chapters 4 and 6 to evaluate the movement segmentation approaches. Parts of the recorded datasets were automatically segmented using the vMCI algorithm introduced in chapter 4 with a manual annotation. These datasets are subsequently used in chapter 5 to evaluate different movement recognition approaches. The recording modalities and each recorded dataset are described in detail in this chapter. Parts of these descriptions and corresponding

name	section	# subjects	# recorded movements	manually segmented	manually labeled
step movements	3.2	6	171	x	x
lever-pulling data	3.3	2	68		x
pick-and-place	3.4	3	18 (+8)	x (partly)	x
ball-throwing	3.5	10	240		x
stick-throwing 1	3.5	7	697		x
stick-throwing 2	3.5	3	34	x	x
gestures	3.6	6	1045		x
object rotation	3.7	3	33	x	x

Table 3.1.: Recorded datasets. for the pick-and-place movements, only 18 recordings were manually labeled.

visualizations are taken or adapted from several publications [Gutzeit and Kirchner, 2016, Gutzeit et al., 2016, Gutzeit, 2021, Gutzeit, 2022, Gutzeit and Kirchner, 2022].

Ethics Statement: All movement data was conducted in accordance with the declaration of Helsinki and approved with written consent by the ethics committee of the University of Bremen. Subjects gave informed and written consent to participate.

3.1. Movement Recording Systems

3.1.1. Qualysis motion tracking

The company Qualisys AB produces motion tracking cameras, which can track marker position very precisely at a high framerate. The data acquired for this work were recorded using Oqus300 cameras, which can track positions of markers with an error of less than 1mm at 500 Hz.

Passive reflecting markers were used, of which the $3D$ positions are tracked using infrared light. To record the arm movements during manipulation tasks, single markers were attached near the shoulder and the elbow of the subjects. Clusters of three markers were attached to the hand and additionally to the back of the subjects. With these orientations can be determined. The orientation of the hand can be an important

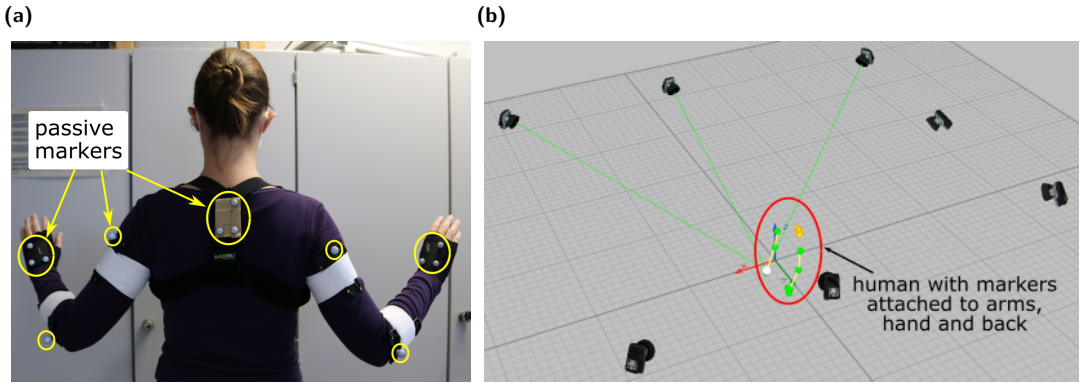


Figure 3.1.: Qualysis motion tracking setup. (a) Participant with passive markers attached to back, arms, and hands (*image adapted from Figure 3 in [Gutzeit, 2022]*). (b) Exemplary camera arrangement. In this setup, positions of the markers are tracked using 7 cameras. The highlighted white marker on the right hand is in this position tracked by 3 cameras, as indicated by green lines.

feature to differentiate between varying manipulation movements. The orientation of the back is used to transfer the recorded data into a coordinate frame local to the subject, which makes the recorded movements independent from the position of the subject in the global coordinate frame of the tracking system. If desired, additional makers can be placed on the hand. For example, with markers on the fingertips not only the hand position but also the grasping movement can be recorded. A marker setup for dual arm movement recordings, for which markers are attached to both arms and hands, can be seen in Figure 3.1(a).

The movements were recorded in a lab in which several cameras are mounted at the walls. Additional cameras can be added to the setup using tripods. The cameras were arranged in a way that during movement recording each marker was tracked by at least 2 cameras. In this way, the system can obtain the three-dimensional positions of the markers. An example camera setup consisting of 7 cameras can be seen in Figure 3.1(b). One marker attached to the right hand of the subject is highlighted. This marker can in this position be seen by three cameras, visualized by green lines. The cameras are arranged below the ceiling around the subject. Before recording, the camera system needs to be calibrated. With a good camera arrangement and calibration, the marker position can be tracked with a measurement error below 1mm. In the experiments conducted for this work, 7 cameras were used for motion tracking

to achieve high precision and minimize occlusions. For the recording of stick-throwing movements described in section 3.5, up to two additional cameras were integrated into the setup. The position and/or orientation of certain points in the environment of the subject or the objects that are manipulated can be tracked by placing additional makers at these positions.

Since passive markers were used for motion tracking, the recorded marker trajectories needed to be assigned to a marker label denoting the marker position (e.g., "right shoulder" denotes the marker placed near the right shoulder). This can be done manually using the Qualisys Track Manager provided with motion cameras [QTM, 2022]. However, this manual marker labeling can become time intensive if the marker positions were not continuously tracked, for example due to a suboptimal camera arrangement or occlusions. In these cases, the recorded trajectories are fragmented and each fragment needs to be assigned to the correct marker. Alternatively, the Qualisys track manager offers an "automatic identification of markers" which uses previously labeled motion data to infer the marker labels.

3.1.2. Xsens motion capture suit

A further system used to record human movements is the MVN Avinda sensor suit of the company Xsens [Xsens, 2022]. The sensor suit consists of 17 IMUs, which are placed to predefined positions on the body, as visualized in Figure 3.2. The IMUs measure angular velocities and accelerations in each direction. Using the provided capturing software, the position and orientation of each body part of the subject can be determined at a maximal frequency of 240 Hz. For this, certain body statistics need to be measured before recording, such as height of the knee, hip, shoulder, and whole body, or the arm span. From these measurements, an internal body representation is determined, as depicted in Figure 3.2(b).

Using the Xsens software, the measurements of the IMUs are directly matched to the internal body representation during recording. This minimizes the problem of drifting positions which often happens in IMU recordings. In contrast to data acquisition using the Qualisys motion tracking system, motion recordings using the IMU sensor suit are not limited to a lab environment that is equipped with cameras. However, the system is less precise compared to Qualisys and the position of, for example, objects which are manipulated cannot be recorded directly using the software of the sensor suit.

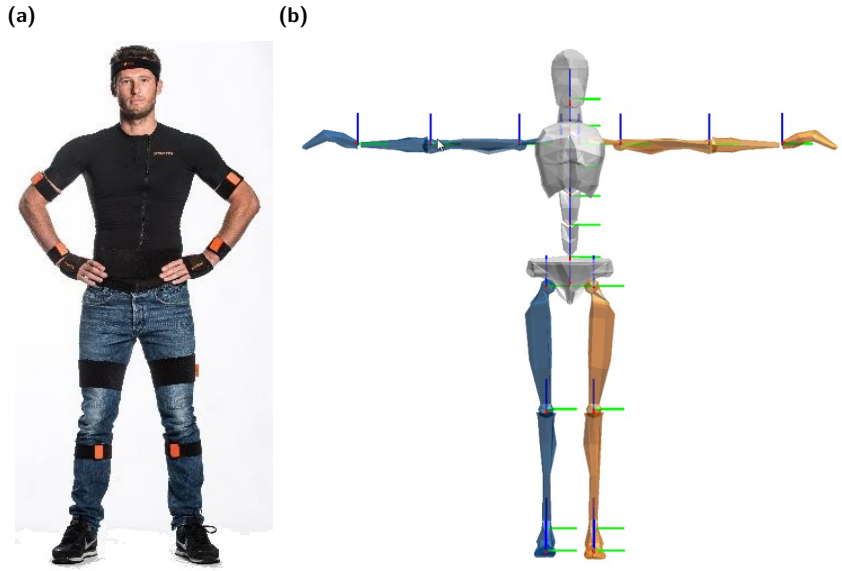


Figure 3.2.: (a) Xsens Avinda sensor suit and (b) internal representation of the human body in the corresponding *MVN Analyse Software*. *Image (a) is taken from the software user manual [Xsens, 2022].*

3.2. Reference Setup

The reference setup was designed to record movements in a restricted environment in which movement segments can easily be identified to generate a ground truth segmentation. Additionally, the setup should be used to analyze different movement features in an environment with only little possibility to vary movement direction. The reference setup is built in two layers: a bottom layer consisting of an iron sheet and a top layer in which a step-like path is cut. Through this path, a stick should be moved by the subject which is connected to the bottom layer via magnetic balls. With this connection the stick can be movement through the path without leaving the plane. The setup can be seen in Figure 3.3. Each step has a height and length of 16 cm, the path a width of 4 cm and the stick a diameter of 3.6 cm.

If the stick is moved through the pattern, the position of the hand is very restricted, as the subjects have only little possibilities to move the stick differently than on the straight line through the pattern. Thus, the subjects are forced to do point-to-point movements. Due to the restricted movement position, the recorded movements can

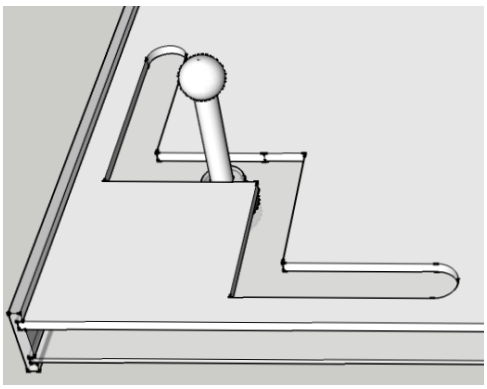


Figure 3.3.: Reference step design. The stick, which is connected to the bottom layer with magnetic balls, should be movement through the step-like path which is cut into the top layer.

be segmented into the main movement building blocks *up*, *down*, *right*, and *left*, with ground truth segment borders at the corners of the step pattern as indicated with circles in Figure 3.4.

Although the position of the stick is very restricted in the step setup, the movement speed can be varied, which results in different velocity profiles of the hand, see Figure 3.4. In chapter 4, the step movements are used to analyze different features of point-to-point movements and to evaluate the developed automatic segmentation technique.

The movement through the step pattern was recorded for 6 different subjects using the Qualisys system. Each subject performed several repetitions of the movement, where the task in one repetition was to move the stick from lower left of the pattern to the upper right and back, resulting in 8 point-to-point movements. This was repeated several times for roughly three minutes with a short break after each minute. The subjects were instructed to perform the movement as precisely as possible, i.e., without hitting the borders while moving as fast as possible. In total, 171 repetitions of the movement where recorded.

3.3. Lever-Pulling Movements

In the lever-pulling demonstrations, the task of each subject was to pull a lever down. The lever was fixed to a table resulting in a strongly predetermined movement execution but compared to the reference setup in a three-dimensional instead of a

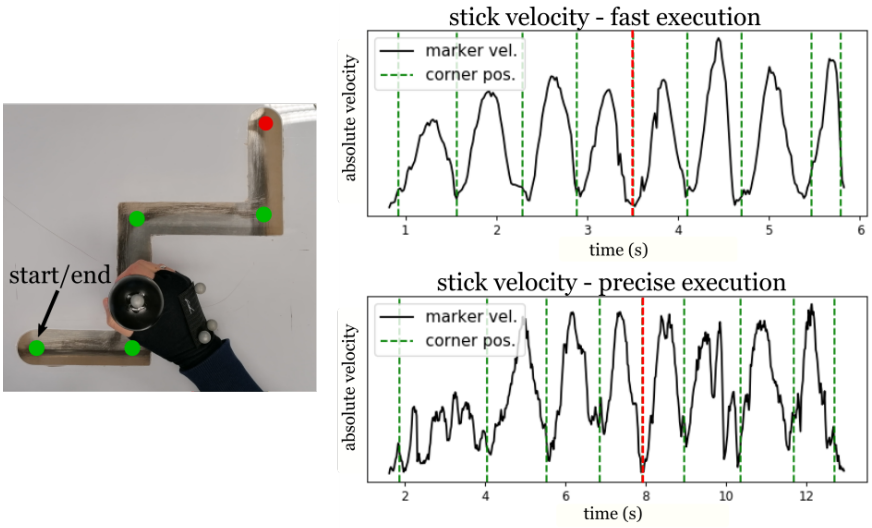


Figure 3.4.: Example of the bell-shaped velocity pattern in human point-to-point movements. The subject was instructed to move a stick though a step pattern, starting at the lower right, going to the upper right and backwards. The absolute velocity of the hand is visualized on the right side. The positions of the corners are marked as green dots in the left image and green dashed lines on the right side respectively. The turning point of movement is indicated in red. The movement was performed fast (right top) and precisely (right bottom), i.e., with the intention to not hit the boundaries. In both cases, bell-shaped curves can be observed in the velocity of the hand for single point-to-point movements. *Image adapted from Figure 1 in [Gutzeit and Kirchner, 2022].*

two-dimensional space. The setup is depicted in Figure 3.5. At the beginning of each movement the subject was in a rest position with the arm hanging down at the side of the body. Next, the subject reached for the lever with the right arm and pulled down the lever. Finally, the subject turned the arm back to the rest position (arm hanging down). After returning to the rest position, the lever had to be pulled up again, which was done with the left arm and was not recorded by motion tracking. This very simple behavior is selected to show that for simple movements only very few demonstrations are needed for movement classification in chapter 5. The individual movement parts of each demonstration could be divided into 4 different main classes: *idle*, *approach forward*, *move lever*, and *move to rest*. In the recordings, only the movement of the arm was tracked and not the position of the involved object, the lever. This is because the spatial distance of the lever to the demonstrator's hand is fixed and plays no role in

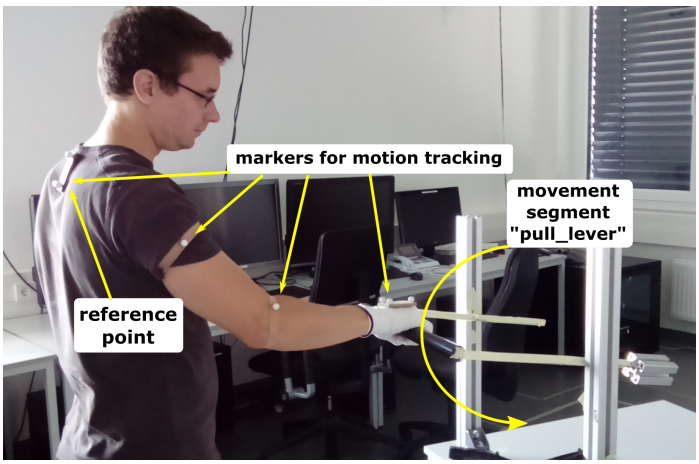


Figure 3.5.: Lever-pulling setup. *Image adapted from [Gutzeit et al., 2016], Figure 3 (b).*

this experiment. The lever-pulling task was demonstrated by two different subjects, performing 32 and 36 pulls, respectively, which were recorded with the Qualisys system.

3.4. Pick-and-Place Movements

Pick-and-place movements of three different subjects were recorded using the Qualisys system, in which the task was to grasp a box from a shelf, place it on a small table on the right side and vice versa. To increase movement variability, the exact position where the box should be placed was not specified. In these movement recordings, markers were placed next to the back, right arm, and right hand of the subjects also on the thumb, index, and middle finger of the right hand. Additional, two markers were placed on the box. The marker setup as well as the performed movements can be seen in Figure 3.6.

After placing the box on the table or the shelf, the subject moved the arm into a resting position. This results in demonstrations composed of 7 different segments assigned to the following movement categories: *approach forward*, *move object to table*, *move to rest right*, *approach right*, *move object to shelf*, *move to rest down*, and the class *idle* for periods in which the subjects did not move their arms. The pick-and-place task was performed by three different subjects, repeated 6 times by each. Two of these subjects performed the task again with four repetitions while their movements were

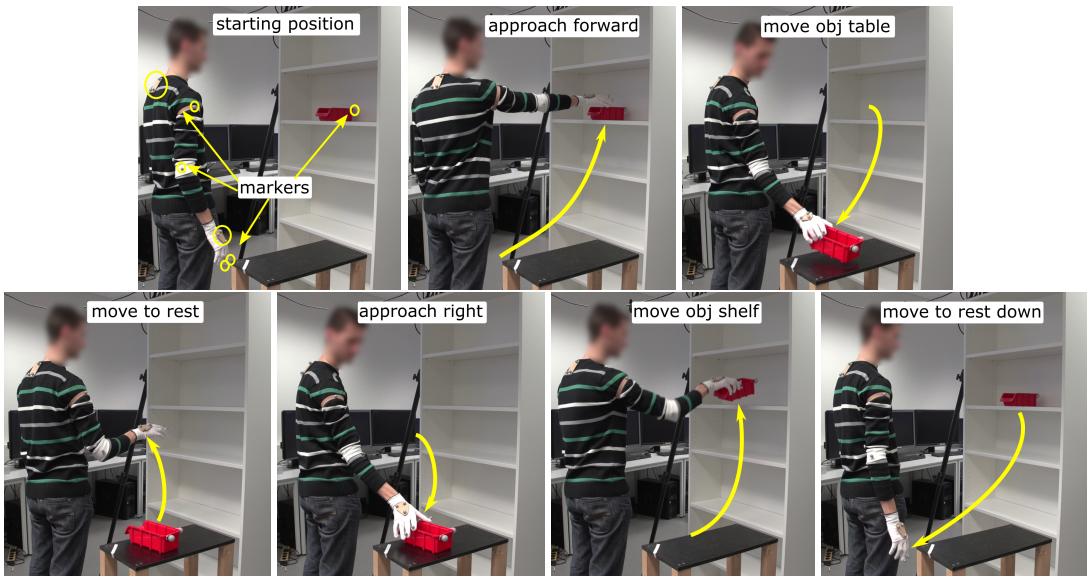


Figure 3.6.: Pick-and-place movement consisting of 7 main movements as depicted in the images. Next to the markers on the back and right arm and hand, additional makers are placed on the tip of the thump, index, and middle finger as well as on the manipulated box.

recorded with slightly different camera positions and a different global coordinate system. This resulted in different positions of the person and the manipulating object in the scene which should be handled by the presented movement segmentation and recognition methods.

3.5. Throwing Movements

Compared to the point-to-point, lever-pulling, and pick-and-place movements discussed in the previous sections, throwing movements show different patterns, as they are faster and the hand must be moved with a certain velocity in order that the object that its thrown reaches its goal position. This makes throwing movements especially useful to evaluate imitation learning methods. Ball- as well as stick-throwing movements were recorded using the Qualisys system.

The ball-throwing movements were recorded using seven motion tracking cameras. Five of these cameras were focused directly on the subject, the rest were focused on the target area. This setup is chosen to also record the movement execution on a robotic

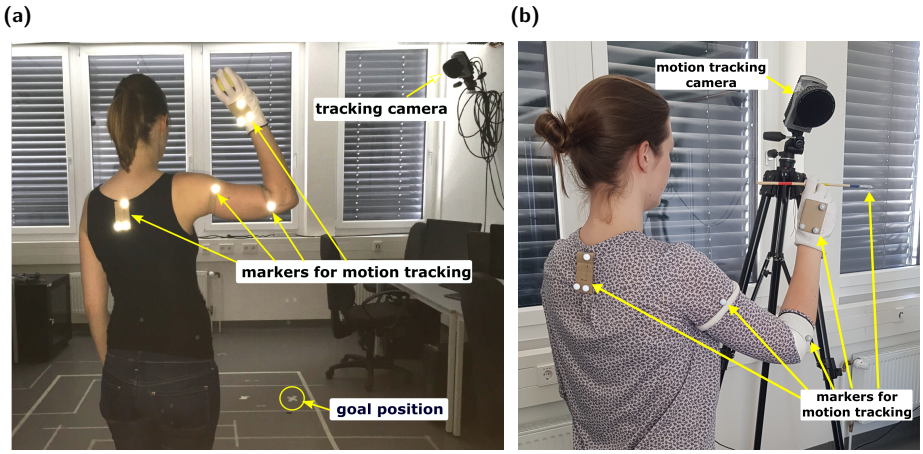


Figure 3.7.: Setups to record ball-throws (a) and stick-throws (b). *Images adapted from [Gutzeit and Kirchner, 2016], Figure 3, and [Gutzeit et al., 2019], Figure 2 (a).*

arm after movement imitation, in which the thrown ball is additionally tracked to determine the goal position. The subjects had to throw a ball to a goal position on the ground, approximately 2 m away, as depicted in Figure 3.7(a). To limit the range of possible throws, they were instructed to throw the ball from above, i.e., the hand is above the shoulder while throwing (see Figure 3.7(a)). Between the throws, the subjects should move their arm to a resting position in which it loosely hangs down. Throwing movements of 10 subjects were recorded. All subjects were right-handed and had different throwing skills ranging from non-experts to subjects performing ball sports in their free time, like basketball, volleyball, or handball. Each throw can be split into 3 main movement phases: the *strike out* movement at the beginning, the actual *throw*, and a *swing out* movement at the end. In between short phases of no movement, in which the subject is *idle*, can be observed. Each subject demonstrated 8 throws in 3 experiments which results in total numbers of 24 throws per subject, 30 experiments, and 240 throws for all subjects.

Additionally, two datasets of stick-throwing movements in a Touhu scenario were recorded. Touhu, also known as pitch-pot, is a throwing game that is traditionally played in Eastern Asia. The goal is to throw a stick from a given distance into a pot. The main movement parts are with *strike out*, *throw*, and *swing out* the same as in the ball-throwing scenario. However, the execution of the *throw* differs, because the stick

Figure 3.8.: Main throwing segments. Shown are the positions of the markers placed on back, right arm, and right hand. The lines show the movement, the end position of each marker is marked with a dot. The three hand markers are highlighted in blue.

is grasped differently compared to grasping a ball, see Figure 3.7. Stick-throwing demonstration of 7 different subjects were recorded using 9 motion tracking cameras. Each subject performed between 41 and 246 throws. In total, 697 throws were recorded.

In the second stick-throwing dataset, an additional maker was placed on the stick to determine its position during the throw. These recordings will be used to compare the trajectories of the stick in the throwing demonstration to throwing movement imitation by a robotic system in chapter 7. In this second stick-throwing scenario demonstrations of three subjects were recorded using 8 motion cameras. The subjects performed 10, 11, and 13 throws respectively.

3.6. Gestures

The gesture dataset was recorded using the Xsens sensor suit at 60 Hz and is used in chapter 5 to compare different movement recognition methods. Eleven gestures, which may be relevant in human-robot collaboration, were recorded. All gestures were performed with the right arm and started in a neutral position, in which the arm hangs relaxed next to the body. The following gestures, which are depicted in Figure 3.9, were recorded:

- *come closer*: The hand is waved towards the body. At the beginning of the movement, the palm faces upward, then the wrist is bent towards the palm and the elbow is slightly bent. The waving movement can be repeated several times.
- *move backwards*: The hand is waved away from the body, with the back of the

Figure 3.9.: Recorded gestures. Arrows indicate the direction of the movement. The performed gesture from top left to bottom right are: *come closer*, *move backwards*, *move upwards*, *move downwards*, *move left*, *move right*, *stop*, *rally*, *hello*, *thumbs up*, and *thumbs down*. *Image extracted from [Gutzeit, 2021], Figure 2.*

hand rotated towards the body of the subject (opposite to *come closer*).

- *move upwards*: The flat hand with the back of the hand rotated downwards is waved several times upwards.
- *move downwards*: The back of the hand is rotated upwards while the hand is moved several times downwards (opposite to *move upwards*).
- *move left*: With the loose hand and the arm stretched forward, a wiping gesture from right to left performed with the wrist of the hand.
- *move right*: Wiping gesture from right to left performed with the wrist of the hand (opposite of *move left*).
- *stop*: The subject raises the arm, which the palm of the hand stretched away. This position is kept for a short time period before moving the arm back.
- *rally*: The hand is raised to head height, the index finger is extended upwards, and the remaining fingers are closed into a fist. The hand is then moved several times in a circular path without changing its orientation.
- *hello*: The flat hand is raised to about head height and waved to the right and

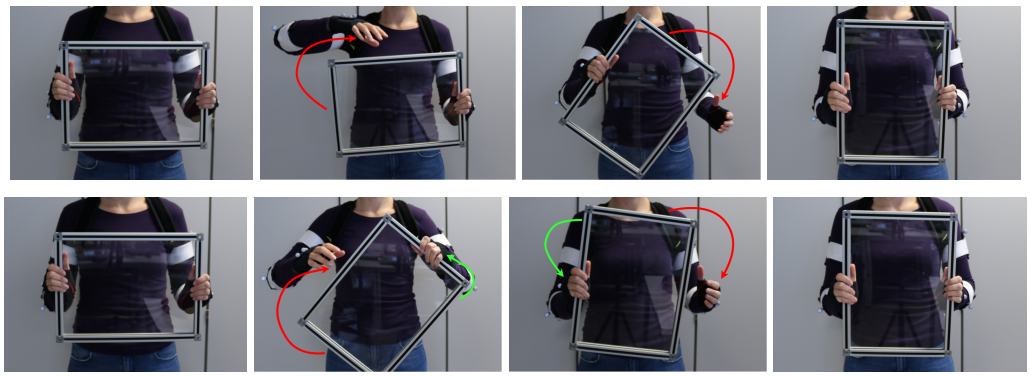


Figure 3.10.: Movement steps to rotate a rectangular frame clockwise. In the top row, the rectangular frame is turned by first holding the object with the left hand and re-grasping with the right hand, and second turning with the right hand and re-grasping with the left. In the lower row it is turned by first re-grasping with the right hand while turning with the left hand, followed up by a turn of the right hand and a re-grasp of the left. That means, while the same dual arm action *turn clockwise* is executed, two different combinations of building block movements for both hands are used. *Image extracted from [Gutzeit, 2022], Figure 1.*

left. The palm of the hand points away from the subject.

- *thumbs up*: The hand is stretched out in front, the fingers closed into a fist and the thumb extended upwards. The hand remains in this position for a moment before the arm is moved back to the starting position.
- *thumbs down*: Opposite to *thumps up* with hand rotated downwards.

Each gesture was demonstrated one time to the subjects before recording. Afterwards, each subject performed each gesture with the instruction to move naturally. For recurring gestures, such as rally, the number of repetitions was not specified but intuitively selected by the subject. In total, each subject performed each gesture $10 - 11$ times. For one subject between 30 and 50 repetitions of each gesture were recorded. In total, 1045 examples of different gesture executions were acquired.

3.7. Dual arm Object Rotation

The most complex data recorded for this work is a dual arm movement, in which an object is rotated. The participants were instructed to grasp a rectangular frame

of aluminum profiles with acrylic glass in the middle with both hands and move it to a horizontal position. Then, the rectangular frame should be turned into a vertical orientation with re-grasping, as visualized in Figure 3.10, and finally back into horizontal orientation. At the end, the frame is placed back on a table standing in front of the subjects. The participants were not restricted in how they move their hands during object rotation as long as the object turning starts and ends in a horizontal position. They intuitively decided if they wanted to turn the rectangular frame first clockwise or counterclockwise. Between 10 and 13 repetitions of turning the frame were recorded from 3 different participants. The data consists of four different dual arm actions *lift*, *turn clockwise*, *turn counterclockwise*, and *place*, with 7 building block segments for each hand: *lift*, *re-grasp down*, *re-grasp up*, *hold*, *turn clockwise*, *turn counterclockwise*, and *place*. To rotate the rectangular frame, different combinations of these building block movements were performed by the subjects. As an example, two variants of turning the frame clockwise are depicted in Figure 3.10. These movements are used in chapter 6 to evaluate the hierarchical segmentation. In total, 33 examples of each action were recorded, with different compositions of building block segments for each action.

3.8. Manual Segmentation

In order to analyze different features of the recorded movements and to evaluate the automatic segmentation, a ground truth segmentation is needed. Whereas this ground truth segmentation comes naturally with the position of the stick in the pattern in the step data, it becomes more difficult to determine exact segment borders with increasing movement complexity. To determine segment borders for the recorded movement trajectories manually, a time series visualization tool was implemented. With this tool, the 3D positions of the markers placed on the human can be visualized as time series in a two-dimensional plot, which is synchronized to a 3D visualization of the marker positions. The tool allows to directly define segmentation points in the time series by clicking on the time frame at which a segment border is assumed.

The visualization of stick-throwing movements using this tool can be seen in Figure 3.11. In plot (c) of this figure, the velocity of the marker(s) of interest for the whole movement demonstration can be seen as well as the summed-up velocity of all recorded markers, whereas the velocity of the selected marker is scaled. In the other

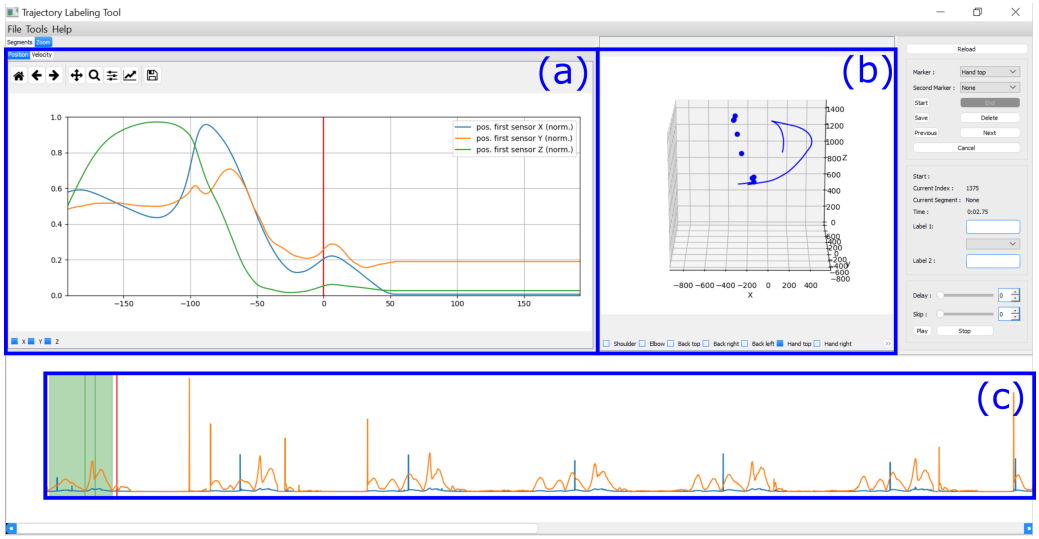


Figure 3.11.: Trajectory labeling tool. In the depicted screenshot, a sequence of stick-throwing movements should be manually segmented. Three segments are already defined, as visualized in green in plot (c) in which the cumulative velocity of all markers is shown in blue and the velocity of the selected marker `Hand top` in orange. By moving the red vertical line, the starting or end point of the next segment can be selected. The marker trajectories are shown in detail in plot (a), in this example, the position of the selected marker is shown. In plot (b), all marker positions are visualized in a three-dimensional plot as well as part of the trajectory of the selected marker `Hand top`.

visualizations in the tool, the three-dimensional marker positions can be seen (plot (b)) as well as the position and velocity of the selected marker(s) in a detail plot (a). In this visualization, it can be selected if the velocity or the position should be shown in detail. By moving the slider in plot (c) with the mouse, the user can go through the trajectory to understand the movement. If the slider is moved to a position where the user identifies the start point of a new movement, the button `start` should be pressed. At the end of the segment the button `end` should be clicked. The label of the defined segment can also be defined.

By using this labeling tool, the step movements, 18 of the pick-and-place movements, the second stick-throwing recordings as well as the dual arm object rotation movements were manually segmented by the same person to generate a ground truth for the evaluations in this work. Furthermore, it is possible to load segmentation borders determined by the vMCI segmentation approach into the labeling tool. With

this, the segmentation points can be verified and if needed corrected. Additionally, labels can be assigned to the segments, which was done for all datasets.

Chapter 4.

4 Segmentation into Building Blocks

Human movement segmentation is a challenging problem because in naturally performed behavior, the variations in the execution of the same movement can be big, even if it is performed by the same person. By using automated segmentation approaches, which can handle inter-subject as well as intra-subject variations, applications which require segmented movement data, such as learning from demonstration or other applications in human-robot interaction, can be sustainably simplified and accelerated. In this chapter, the velocity-based Multiple Change-point Inference (vMCI) algorithm is introduced which identifies building blocks in human manipulation movements in an unsupervised manner. With this algorithm, an approach is presented which uses repetitive characteristics in the velocity of the hand in human arm movements, as described in section 2.3, directly to segment a movement sequence into its building blocks. Before the description of the algorithm, the characteristics of the movements that were recorded within this thesis are examined in more detail. Different features of point-to-point movements as well as pick-and-place movements are compared to verify the assumption that a bell-shaped velocity profile is a suitable characteristic of building blocks in manipulation movements.

To automatically detect building blocks in recorded data streams, there are different requirements for the segmentation algorithm. First, it should be able to handle variations in the movement data. Although, the CNS always executes very similar movements, as discussed in section 2, several repetitions of the one movement can

show considerable variations in the movement trajectory. In a grasping movement, this could result, e.g., from changing relations in the position between the hand and the object at the start of the movement. This variation is even higher if the same movement is executed by different persons which origins, for instance, from different limp sizes of the persons. Also the velocity can show variations. For example, the velocity and the start or end of the movements can differ, especially if a sequence of movements is performed. Additionally, the maximal points or the width of the velocity curve can differ between movement executions. If a movement is performed slowly, the velocity also may contain noise. Additional noise in the movement can result from inaccuracies in the movement recording induced by the recording system. However, this can be neglected for precise systems like the Qualisys motion capture system. All these variations in the movements make it more difficult to automatically segment recorded data reliably.

Next to the handling of movement variations, it is desirable that an automated segmentation approach is applicable with no or minimal adaptions, e.g., of parameters, on different movements performed by different persons. This reduces the tuning of parameters or pre-knowledge which may be required. Furthermore, the development of online segmentation approaches makes it easier to integrate the methods into an interaction framework running on a robotic system.

To detect building blocks automatically in human manipulation movements, the probabilistic and unsupervised segmentation method vMCI is introduced. This algorithm is based on Bayesian inference, where noise and variations in the recorded movement can be integrated into the model. The segmentation algorithm is based on an online variant of an algorithm to detect change-points in time-series data proposed in [Fearnhead and Liu, 2007] which is called Multiple Change-point Inference (MCI). Within this thesis, this approach was extended to a velocity-based MCI (vMCI), in which the characteristics in the velocity profiles, as described in section 2.3, are integrated into the segmentation process. VMCI is the first unsupervised behavior segmentation algorithm, which includes the characteristic velocity pattern into the segmentation process to detect building blocks in manipulation movements. The developed algorithm is evaluated on artificial data and on several real human movements in comparison to the original MCI algorithm, a method based on local minima segmentation (locMin) and two promising probabilistic segmentation methods from the literature: Beta-process auto-regressive Hidden Markov Model (BPARHMM) [Fox

et al., 2009] and Probabilistic Segmentation (ProbS) [Lioutikov et al., 2017].

This chapter is structured as follows: First, a short overview of related work is given, which is followed by the analysis of movement features in point-to-point as well as pick-and-place movements. Afterwards, the vMCI algorithm is presented in section 4.3 which is followed by several experimental evaluations of vMCI with respect to influence of the hyper-parameters of the algorithm on artificial data and in comparison to MCI, locMin, BPARHMM, and ProbS on several real movements. At the end of this chapter, the results are discussed. The texts, figures, and tables in this chapter are partly taken or adapted from [Senger et al., 2014] and [Gutzeit and Kirchner, 2022].

4.1. Related work

For a long time, most approaches to segment time series data of human movements were supervised and required manual segmentation of movement examples to generate training data. An overview is given in [Aggarwal and Cai, 1999]. In the last years, several new approaches were proposed which can be run unsupervised [Lin et al., 2016], i.e., manual efforts can be reduced and the methods are also applicable if the behavior segments are not know in advance. [Fod et al., 2002] presented a heuristic approach, in which segment boundaries are detected using the angular velocities of several degrees of freedom. If these cross zero, the start of a new movement segment is assumed. This approach is very sensitive to noise and tends to over-segment the data, in particular if many DOFs are considered as data input. Similarly, in [Jenkins and Matarić, 2003] segments are detected based on 'swings' in the velocity of joint angle data. In both approaches, thresholds need to be defined in advance, which probably have to be adapted for different datasets.

Probabilistic approaches generalize to different movement executions by integrating movement variations directly into the model. With a predefined number of segments to be detected, the probabilistic approach presented in [Chiappa and Peters, 2010] detects movement identities in table tennis demonstrations using a belief network. This approach is applicable without manual segmentation of training data, but the number of segments must be determined in advance or by, e.g., cross-validation. Another probabilistic model, which does not require a priori knowledge, was presented in [Kulić et al., 2012]. In that work, human movements are segmented and additionally

clustered based on HMMs. Gong et al., on the other hand, used Hilbert space embedding of distributions for segmentation [Gong et al., 2012]. In both works whole-body motions were segmented online.

A probabilistic model, for which also the implementation is available online, is presented in [Fox et al., 2009]. In this method, repeated building blocks of the same movement are inferred using the so-called *beta process autoregressive HMM* (BPARHMM). Like in the method proposed in this thesis, the segments are represented by linear regression model (LRM)s. Their inference model allows for shared regression models, i.e., repeated building blocks of the same movement can be detected. Niekum et al. used the BPARHMM algorithm to segment kinesthetic demonstrations provided by hand-coded controllers of a pick-and-place task [Niekum et al., 2012]. The MCI algorithm [Fearnhead and Liu, 2007], which serves as a starting point for the method proposed in this chapter, is a similar segmentation method which works without a priori knowledge of the number of segments. Furthermore, it has a data model that allows to integrate characteristic velocity patterns into the segmentation which is crucial to find segments related to re-usable building blocks, as discussed in chapter 2. The MCI algorithm was used by Konidaris et al. to segment trajectories recorded by maneuvering a robot through a corridor [Konidaris et al., 2012]. In [Lioutikov et al., 2017], a probabilistic segmentation approach called ProbS is presented, in which segmentation points are inferred using expectation maximization based on initially over-segmented data. The detected segments are represented using DMPs [Ijspeert et al., 2013], which is a popular representation of movement demonstrations in LfD. Using this approach, it was possible to automatically learn primitive movements that are needed to assemble a chair with a robotic system from human demonstrations. In [Lioutikov et al., 2017], ProbS performed better than BPARHMM on the examined datasets.

Most of the methods presented in the literature are evaluated on small datasets of single subjects [Lin et al., 2016]. The approach presented in this chapter is evaluated on multiple datasets, each containing movement data of different subjects to show the generalization ability of the algorithm. Furthermore, the presented segmentation algorithm is the only approach which includes characteristic velocity pattern of building blocks into the segmentation process. It is compared to MCI, BPARHMM, and ProbS, which are the most promising probabilistic segmentation approaches presented in the literature.

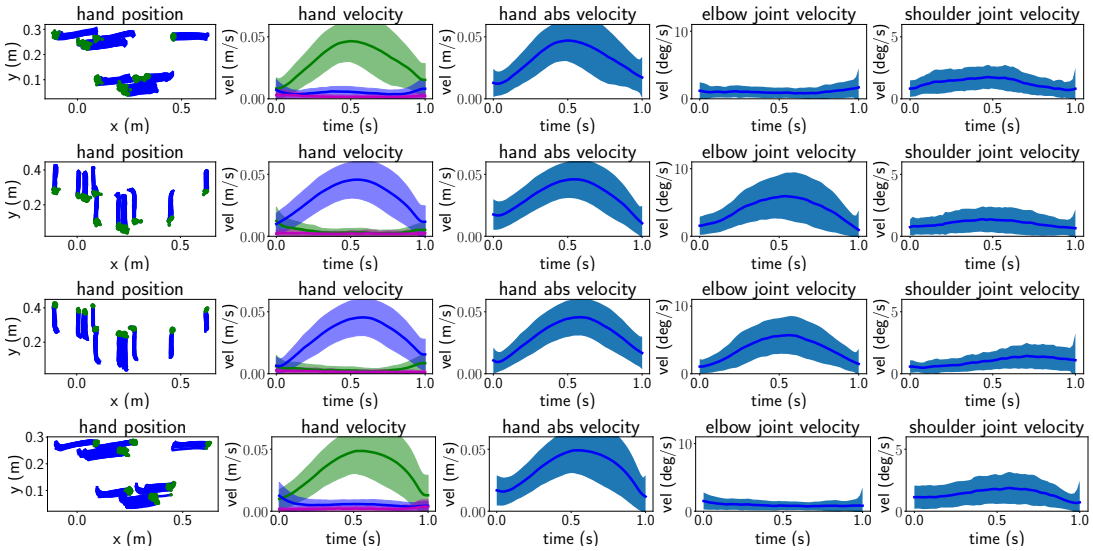


Figure 4.1.: Different features of the point-to-point movements in the reference testbed shown in Figure 3.4. Shown is the mean value and standard deviation of the point-to-point movements *right*, *up*, *down*, and *left* (columns 1-4) for the movement features: hand velocity in x, y, and z direction (lines colored in green, blue, and magenta), absolute hand velocity, elbow joint velocity, and shoulder joint velocity. In the first column the positions of all analyzed segments are plotted. The only feature that looks similar for all movements is the absolute hand velocity, which is a bell-shaped curve. *Image extracted from [Gutzeit and Kirchner, 2022], Figure 4.*

4.2. Characteristics of Building Blocks

In chapter 2, several experiments from the literature were presented which indicate that in human arm movements bell-shaped curves in the velocity of the hand are a characteristic feature of movement segments [Morasso, 1981, Morasso and Mussa-Ivaldi, 1982]. The data recordings of the reference step pattern and the pick-and-pace data were used to examine this theory on movement data recorded at high precision. Using the position-based visualization described in section 3.8, the point-to-point movements of the step pattern and the pick-and-place movements were manually divided into 4 and 7 movement building blocks as described in section 3.2 and 3.4 respectively. Parts of the movements which could not be assigned to one of the building block classes remained unsegmented and were not considered for further analysis. For all movement segments, the following features were calculated from the

recorded marker positions:

1. hand velocity in x, y, and z direction,
2. absolute velocity of the hand,
3. elbow joint velocity calculated using the temporal derivative of the elbow joint position, which is determined using the angle between lower and upper arm,
4. shoulder joint velocity calculated using the temporal derivative of the shoulder joint position, which is calculated using the angle between upper arm and upper body.

The features of all segments were normalized to the same time period between 0 and 1 seconds. For all segment features belonging to the same movement class, the mean value and its standard deviation were calculated. This mean feature values of each class were visualized to determine similarities and differences in the features of the different manipulation movements.

The resulting plots for the point-to-point movement building blocks are shown in Figure 4.1. In these plots, the 2D position of the hand is visualized as well as the mean values of the four calculated features. Each segment class was executed two times in one demonstration of the step pattern movement. Additionally, the reference setup was located at different positions in the global coordinate frame for individual movement recordings, so that movements belonging to the same movement class can be located at different positions (first column in Figure 4.1). Based on the direction of the movement, the highest velocity can be observed in a different coordinate of the hand position. For some movement classes an increasing elbow joint velocity can be observed. In the *up* and *down* movements, the main change in the velocity is in the y coordinate of the hand and in the elbow joint velocity. A change in the x coordinate of the hand can be observed in the movements belonging to the classes *right* and *left*, without an increasing elbow joint velocity. The only feature that looks similar for all movement segments regardless of the position and direction of the performed movement is the absolute velocity of the hand.

The visualization of the features for the pick-and-place movements can be seen in Figure 4.2. In comparison to the point-to-point movements, these movements are three-dimensional and not restricted in the position of the hand. The participants were free in the selection of the exact position to place the box on the table or shelf.

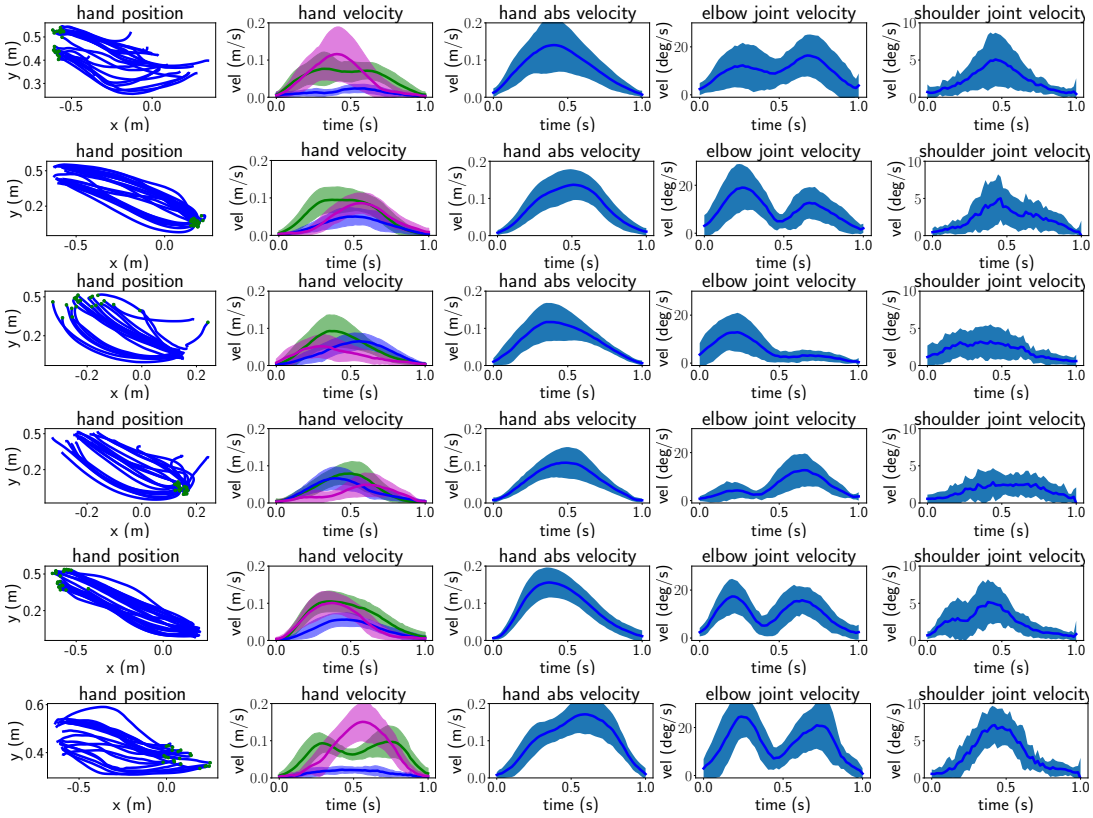


Figure 4.2.: Different features of the pick-and-place movements shown in Figure 3.6. Shown is the mean value and standard deviation of the pick-and-place movements *approach forward*, *move object to table*,*move to rest right*, *approach right*, *move object to shelf*, and *move to rest down*(columns 1-6) for the movement features: hand velocity in x, y, and z direction (lines colored in green, blue, and magenta), absolute hand velocity, elbow joint velocity, and shoulder joint velocity. In the first column the positions of all analyzed segments are plotted. The only feature that looks similar for all movements is the absolute hand velocity, which is a bell-shaped curve. *Image extracted from [Gutzeit and Kirchner, 2022], Figure 5.*

This resulted in a higher variety in movement execution. Nonetheless, the common feature for all building blocks of the pick-and-place task is the bell-shaped pattern in the absolute velocity of the hand, whereas the angular velocity of the elbow and shoulder joint differ between building block movements. Especially the elbow joint velocity shows multiple minima within one movement class, which would result in an over-segmentation if segment borders would be assumed at points where this angular velocity is minimal. However, compared to the simple point-to-point movements, the absolute hand velocity shows more variations for different building blocks in the pick-and-place movements with the main impact that the velocity peak is for some movement classes clearly shifted to one side, for example for the *move object to shelf* class. These variations in the bell-shaped absolute velocity will be considered in the vMCI algorithm.

These observations support the assumption that manipulation building blocks may be characterized using the absolute velocity of the hand. As shown in the next section, segmentation based on these findings improves segmentation accuracy.

4.3. Velocity-based Multiple Change-point Inference

The vMCI is an algorithm which automatically detects the borders of building blocks with a bell-shaped velocity in human movement data. It is based on the MCI algorithm introduced in [Fearnhead and Liu, 2007] and presented in this section.

4.3.1. Data representation

In the vMCI algorithm, it is assumed that a data sequence $y = (y_1, .., y_T)$ of length T, with $y_i = (y_i^p, y_i^v) \in \mathbb{R}^{d+d^v}$ being an observation at time point i with position information of the observation y_i^p and its velocity y_i^v, consists of an unknown number of segments with a bell-shaped velocity. The segments are modeled independently of each other with LRMs. In this manner, they are represented as a weighted sum of basis functions with added noise. The position information of a single segment $y_{i+1:j}$, starting at time point $i+1$ and ending at time point j, is represented as:

$$y_{i+1:j}^p = \sum_{k=1}^{q} \beta_k \phi_k + \varepsilon, \tag{4.1}$$

with q basis functions $\phi_k, k \in \{1, ..., q\}$, model parameters $\beta = (\beta_1, ..., \beta_q)$, and independent and identically distributed Gaussian noise ε with zero mean and variance Σ. This model is identical to the data model in the MCI algorithm [Fearnhead and Liu, 2007]. To infer the model from the data, prior distributions are set over the weights and the noise variance. The parameters β are assumed to be matrix-normal distributed with zero mean and covariances D and Σ along rows and columns respectively. To ensure direct calculation of the posterior probability of the data, conjugate priors are assumed. This results in an inverse Wishart prior for the variance Σ with hyper-parameters ν and S.

To account for the bell-shaped velocity for each segment, the velocity information y_i^v, which is the absolute velocity in the direction of the movement, is in contrast to the MCI algorithm modeled separately in vMCI with a basis function ϕ_v that represents the bell-shaped structure. The basis ϕ_v is defined as a single radial basis function with center c and width r:

$$\phi_v(x_t) = \exp\left\{-\frac{(c - x_t)^2}{r^2}\right\}. \tag{4.2}$$

The width parameter r is chosen to be half of the assumed segment length, i.e., $r = (j - i)/2$, so that the whole segment can be covered by the model. The center c regulates the alignment to different velocity curves. For example, a center location closer to the starting point of the segment allows to approximate a segment with high velocity at the beginning and rather low velocity at the end.

With this, the velocity y_i^v is represented using:

$$y_{i+1:j}^v = \alpha_1 \phi_v + \alpha_2 + \varepsilon^v, \tag{4.3}$$

with weights $\alpha = (\alpha_1, \alpha_2)$ and noise ε^v. Again, weights and noise are matrix-normal distributed with $\alpha \sim \mathcal{MN}(0, D_v, \Sigma_v)$ and $\varepsilon \sim \mathcal{MN}(0, I, \Sigma_v)$. D_v and Σ_v are the prior parameters. The prior distribution of Σ_v is again chosen to be inverse Wishart, $\Sigma_v \sim \mathcal{IW}(\nu_v, S_v)$ to provide conjugate priors. The model order is fixed to 2, with the two basis functions ϕ_v and 1. The constant, weighted with α_2, is added to account for velocities unequal to zero at start or end of the segment.

To determine the emission probability (see Figure 4.3), the likelihood of the data $y_{i+1:j}$ given model m and velocity model m_v needs to be determined. As independent models are assumed for the velocity and the position, the likelihood of the data sequence $p(y_{i+1:j}|m, m_v)$ given the model m of order q and velocity model m_v, can be

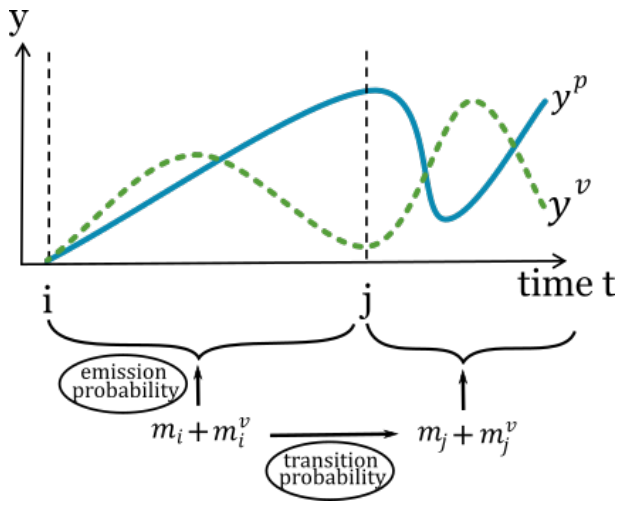


Figure 4.3.: Schematic presentation of the hidden markov model to detect segment borders with vMCI. The observable data sequence y, consisting of position y^p and velocity y^v, is generated by hidden models m and m_v separately for each segment. Segment borders are detected at positions where the underlying models change, either to new models or the same with different parameters. *Image extracted from [Gutzeit and Kirchner, 2022], Figure 2.*

derived by marginalizing out the model parameters β and α, i.e.,

$$\begin{aligned} p(y_{i+1:j}|m, m_v) &= \int p(y^p_{i+1:j}|\beta, m)p(\beta)\,d\beta \cdot \int p(y^v_{i+1:j}|\alpha, m_v)p(\alpha)\,d\alpha \\ &= \int\int p(y^p_{i+1:j}|\beta, \Sigma) \cdot p(\beta|D, \Sigma) \cdot p(\Sigma|\nu, S)\,d\Sigma\,d\beta \\ &\quad \cdot \int\int p(y^v_{i+1:j}|\alpha, \Sigma_v) \cdot p(\alpha|D_v, \Sigma_v) \cdot p(\Sigma_v|\nu_v, S_v)\,d\Sigma_v\,d\alpha. \end{aligned} \tag{4.4}$$

Due to the chosen conjugate priors for both LRMs, the integrals can directly be solved resulting in

$$\begin{aligned} p(y_{i+1:j}|m, m^v) = (2\pi)^{-\frac{nd}{2}} \frac{|M|^{\frac{d}{2}}}{|D|^{\frac{d}{2}}} \frac{|S|^{\frac{\nu}{2}}}{|(y^p)^T P y^p + S|^{\frac{n+\nu}{2}}} \frac{\Gamma_d(\frac{n+\nu}{2})}{\Gamma_d(\frac{\nu}{2}) 2^{\nu d/2}} \\ \cdot (2\pi)^{-\frac{nd_v}{2}} \frac{|M_v|^{\frac{d_v}{2}}}{|D_v|^{\frac{d_v}{2}}} \frac{|S_v|^{\frac{\nu_v}{2}}}{|(y^v)^T P_v y^v + S_v|^{\frac{n+\nu_v}{2}}} \frac{\Gamma_{d_v}(\frac{n+\nu_v}{2})}{\Gamma_{d_v}(\frac{\nu_v}{2}) 2^{\nu_v d_v/2}} \quad , \end{aligned} \tag{4.5}$$

with $M = (H^T H + D^{-1})^{-1}$, $P = I - HMH^T$ used to determine the model evidence for the position data y_p and $M_v = (H_v^T H_v + D_v^{-1})^{-1}$, $P_v = I - H_v M_v H_v^T$ used to determine

the model evidence for the velocity data y_v. In these notations H and H_v refer to the matrices of basis functions, i.e., $H = (\phi_1, ..., \phi_q)$ of shape $n \times q$ and $H_v = (\phi_v, 1)$ of shape $n \times 2$, with n defining the length of the segment, i.e., $n = j - i$. The matrix I is the $n \times n$ identity matrix and Γ_d is the d-dimensional Gamma function. The derivation of this formula can be found in Appendix A.

4.3.2. Online inference of change-points

Using the model likelihood $p(y_{i+1:j}|m, m_v)$, the segmentation points in the data y as well as the underlying LRMs of the observed data can be determined using an online Viterbi algorithm presented in [Fearnhead and Liu, 2007] which will be described in this section. Here, a segmentation point is named change-point and refers to a time point i in the time series y where the underlying LRM changes. The segment models are assumed to be independent of each other and the change-point positions are modeled via a Markov process, as depicted in Figure 4.3. The transition probabilities dependent on the segment length between two change-points and are defined as:

$$P(\text{next change-point at j}|\text{change-point at i}) = g(j - i), \tag{4.6}$$

where $g(l)$ is the probability of a segment having length l. The cumulative distribution function of this length is given by $G(l) = \sum_{k=1}^{l} g(k)$. As proposed in [Fearnhead and Liu, 2007], a geometric distribution for $g(l)$ is assumed, so that $g(l) = (1 - p)^{l-1}p$ and $G(l) = 1 - (1 - p^l)$. Using these distributions, the parameter p regulates the expected segment length, which is $1/p$.

For each time point t, the most likely change-point position j prior to t and the most likely model of this segment from j to t is calculated using an online Viterbi algorithm. This is done by determining the posterior probabilities for each segment which ends at t and for all data models. The algorithm calculates for each $t > 0$, $j = 0, ..., t - 1$, and every model $m, m_v \in \mathcal{M}$:

$$\begin{aligned} P_t(j, m, m_v) = (1 - G(t - j - 1))p(y_{j+1:t}|m, m_v) \\ \cdot p(m)p(m_v)P_j^{MAP}, \end{aligned} \tag{4.7}$$

and

$$P_t^{MAP} = \max_{j,m} \left(\frac{P_t(j, q)g(t - j)}{1 - G(t - j - 1)} \right). \tag{4.8}$$

Equation (4.7) gives the probability that the most recent change-point prior to t occurs at time j with models m and m_v for the segment $y_{j+1:t}$ that has a length of at least $t-j$. The first term is the probability that the assumed segment starting at $j+1$ has a length of at least $t-j$. It is multiplied with the marginal likelihood of that segment having models m and m_v, $p(y_{i+1:j}|m, m_v)$, times the prior probability of the models, $p(m)$ and $p(m_v)$. The last term P_t^{MAP} denotes the most likely change-point position prior to j. In (4.8) the most probable j, m, and m_v are determined. The initial P_0^{MAP} is chosen to be $1/|\mathcal{M}|$. Because the probabilities $P_t(j, m, m_v)$ are very close to zero for most of the possible segments, a particle filter as proposed in [Fearnhead and Liu, 2007] is used to reduce computation time.

4.3.3. Open source implementation

In order that the vMCI algorithm can be used by other researchers, an open source implementation written in python can be found online: `https://github.com/dfki-ric/vMCI_segmentation/`. The code includes two usage examples: One example uses synthetically data consisting of two concatenated DMPs as used in the experimental evaluation presented in section 4.4.1. In the second example, the vMCI implementation is applied on real human motion data. For this, the pick-and-place demonstrations presented in section 3.4 are part of the open source implementation. With this, parts of the experiments performed in section 4.4.5 can be reproduced.

4.4. Experiments and Results

Several experiments were performed to evaluate the performance of vMCI and to compare it to other state-of-the-art segmentation algorithms. In general, the comparison of different behavior segmentation algorithms is difficult because a ground truth segmentation is not available in natural human movements. For this reason, it is desirable that the influence of the parameters of the algorithm is low such that they can be fixed or calculated from the data because an optimization, e.g., via cross-validation, is only possibly if manually segmented data which can serve as ground truth is available. However, the generation of this manually segmented data can be very time intensive. Hence, the reduced parameter influence of vMCI is shown in a first experiment in comparison to the original MCI and the BPARHMM presented in [Fox et al., 2009],

which is one of the state-of-the-art segmentation algorithms present in the literature with promising results [Niekum et al., 2012]. The experiment was performed on synthetic trajectories consisting of two segments with known ground truth.

In the second set of experiments, human movements recorded with the Qualisys motion capture system described in chapter 3 were automatically segmented using vMCI. In these experiments, the capability of vMCI to segment a demonstration into behavior building blocks characterized by bell-shaped velocity patterns is shown. On three different datasets of different complexity, vMCI was evaluated in comparison to MCI, locMin, BPARHMM, and to the ProbS algorithm, which is another promising segmentation technique published recently in [Lioutikov et al., 2017]. The algorithms were first evaluated on movement recordings in the restricted environment of the step reference setup, consisting of point-to-point movements with only little variation in the position between different recordings but with noisy velocity patterns. In further experiments, the ability of vMCI to scale to free human movements is shown on stick-throwing and pick-and-place demonstrations. The algorithms are compared using the F1-score, which is the harmonic mean of precision and recall. Additionally, the number of true positives (TP) and false positives (FP) is investigated in more detail to deduce how many of the segments contained in the data are correctly detected by the methods. In all experiments, the basis functions ϕ of the LRM in vMCI, MCI, and BPARHMM were chosen to be autoregressive with order $q = 1$, i.e., $\phi(x_t) = y_{t-1}$. For this, the data was preprocessed to a mean of zero and such that the variance of the first order differences of each dimension is equal to 1. This preprocessing was not done for the velocity dimension if vMCI was used. The parameter for the distribution of the segment length p in the MCI and vMCI algorithm was fixed to p=0.02 because it has a small influence on the segmentation results due to the Bayesian model of the algorithm.

4.4.1. Segmentation of sequenced DMPs

In this experiment, vMCI was compared to other unsupervised segmentation methods on synthetic data with different parameter configurations to show the reduced influence of the parameters due to the integration of the velocity into the segmentation process. The synthetic dataset consists of two consecutive *dynamical movement primitives* (DMPs) [Ijspeert et al., 2013]. With this, the ground truth segmentation point

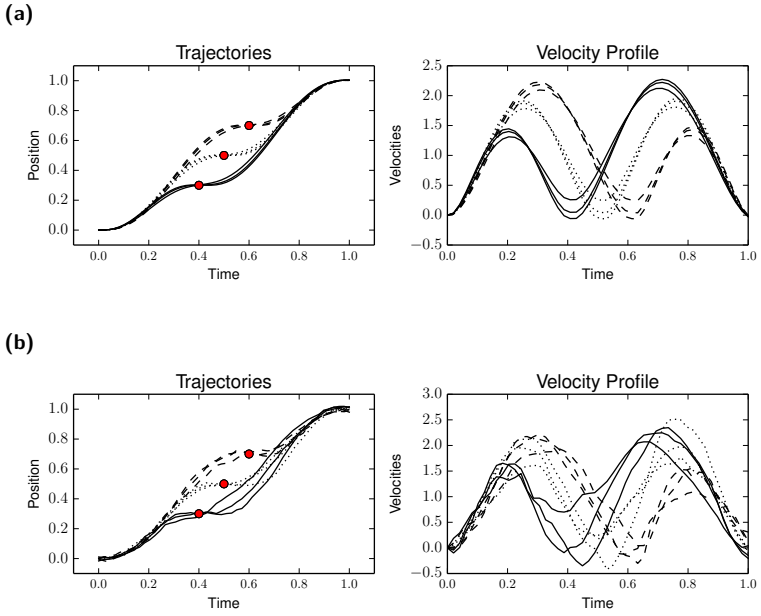


Figure 4.4.: Two datasets of sequenced DMPs. (a) Trajectories of three demonstrations for each of three possible subgoal positions (red points) with their corresponding velocities which were varied at the subgoal positions. (b) Trajectories with corresponding velocities of the second DMP dataset with added noise. *Images extracted from [Senger et al., 2014], Figure 1.*

between the two DMPs is known which makes a comparison of the algorithms possible. DMPs are a popular representation of behavior building blocks in LfD domains. They describe a movement using a dynamical point attractor system, from which arbitrary shape-able, goal-directed movements can be generated.

Using DMPs, two datasets were generated, each consisting of 9 trajectories which are concatenations of two DMPs. The first dataset contains 3 demonstrations for each of 3 different subgoal positions between the two DMPs. In the second dataset, Gaussian noise was added to simulate variance in the movement execution. The DMP sequences have different velocities at the subgoal positions. To generate the trajectories, the DMP representation by Mülling et al. [Mülling et al., 2013] was used where the goal velocity can be modified to a desired value. In Figure 4.4 the generated trajectories are shown with their corresponding velocity.

Using this dataset, vMCI was compared to MCI and to the BPARHMM algorithm.

Table 4.1.: Mean segmentation results on sequenced DMP dataset (with noise).

	F1-measure	number true positives (avg., optimal: 1)	number false positives (avg., optimal: 0)
vMCI	0.97 (0.78)	1.00 (0.86)	0.06 (0.37)
MCI	0.34 (0.34)	0.57 (0.50)	1.17 (0.73)
BPARHMM	0.46 (0.33)	0.26 (0.17)	0.56 (0.52)

The BPARHMM uses the same data model as MCI and has a set of equal parameters. This allows to compare the algorithm with the same set of parameter configurations. An open-source implementation of BPARHMM is available[1].

The three algorithms were compared on the two synthetic DMP datasets with different parameter configurations. The hyper-parameters D, S, and ν, influencing the prior distributions of the model parameter β and the noise variance, of the three segmentation algorithms were varied in the following ranges: $D \in [1, 5, 10, 20]$, $S \in [0.1, 1, 10, 25]$, and $\nu \in [4, 6, 8]$. This results in $4 \cdot 4 \cdot 3 = 48$ different parameter configurations for each demonstration. The other parameters were chosen as suggested in the original publications. Note that due to a more complex inference method, the calculation time of BPARHMM segmenting the synthetic dataset is approx. 20 times larger than the calculation time of the same data with the proposed vMCI.

In Figure 4.5, the position of the obtained segment borders of all parameter configurations and 3 demonstrations are shown in a histogram for each of the segmentation algorithms. The segmentation accuracy measured using the F1-measure and the number of true and false positives for each algorithm for the 9 noisy demonstrations are listed in Table 4.1. The results show that vMCI detected the correct position of the segment transition more accurately than MCI and BPARHMM. Although BPARHMM outperformed the conventional MCI, it showed more false positives than vMCI if the parameters were varied. This means that with the integration of the velocity profiles into the MCI, the segmentation results became more robust with respect to parameter selection. This holds true for different subgoal velocities as well as when adding noise.

[1] `https://emilybfox.su.domains/wp-content/uploads/2021/12/BPARHMMtoolbox.zip`

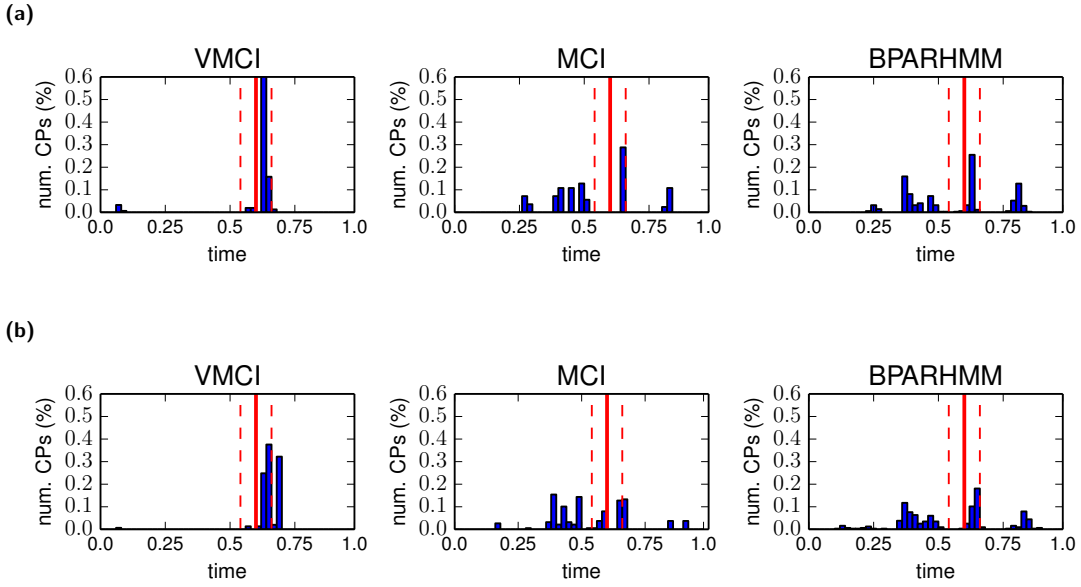


Figure 4.5.: Distribution of detected segmentation points by different algorithms with different parameter configuration. The true segment border is marked with a solid horizontal line. Segmentation points detected in a margin around this point (dashed lines) are treated as correct. As a representative result just segmentation results of trajectories with subgoal position (0.6, 0.7) are shown. (a) Results on DMP dataset. (b) Results on noisy DMP data. *Images extracted from [Senger et al., 2014], Figure 2.*

4.4.2. Selection of hyper-parameters

In the experiments performed on real human movements, the hyper-parameter D, which regulates the variance of the model parameters β along the data dimensions, was set to the identity matrix. This is a good choice because an autoregressive basis is chosen and the data is preprocessed to a variance of one. The hyper-parameters S and ν influence the variance of the weights as well as the Gaussian noise of the LRM along the time dimension. These parameters can directly be calculated from the data to estimate the true variance by determining the variance of the first order differences of the data along the time dimension. As shown in the previous experiment, these hyper-parameters have only little influence on the segmentation result. In the vMCI algorithm, the number of centers was fixed to 3 and the hyper-parameters were calculated identically to the ones of the position LRM from the velocity data.

To run BPARHMM, next to the selection of the basis function and corresponding

Table 4.2.: Mean segmentation results on step pattern movements.

	F1-score	num. TP (opt.: 7)	num. FP (opt.: 0)
vMCI	**0.85**	6.0	**1.1**
MCI	0.63	4.0	1.2
locMin	0.83	**7.0**	3.7
BPARHMM	0.81	6.9	3.1
ProbS	0.18	1.5	1.9

hyper-parameters, several other hyper-parameters had to be set in advance. The same hyper-parameter configurations were used for all datasets based on the suggestions made in [Fox et al., 2009]. The sampling algorithm was run 5 times, with 5000 iterations each. To compare to ProbS, for which no open source implementation is available, the algorithm was reimplemented based on the formulas given in [Lioutikov et al., 2017]. In ProbS, the data is represented using DMPs. Based on an initial over-segmentation the algorithm infers the number and parameters of the DMPs needed to generate the observed data. LocMin was used to generate the initial over-segmentation. Due to the computationally intensive inference steps, the number of iterations was limited to 50 and the algorithm was run separately for demonstrations of each subject. The locMin approach detects a segmentation point at positions where a local minimum occurs in the velocity in a predefined window [Senger et al., 2014]. This window had been varied for each dataset and the margin with the best results was selected for the final evaluation.

4.4.3. Segmentation of point-to-point movements

For the automatic segmentation of the point-to-point data, which contains in the manually determined ground truth segmentation 1368 movement segments, the recorded movements were down-sampled to 30 Hz. In the locMin approach, the window size was set to 0.27 seconds and for pre-segmentation in ProbS the window size was set to 0.13 seconds. All other parameters of the compared algorithms were selected as described in the previous paragraph.

The results of the evaluated algorithms can be seen in Table 4.2, in which the mean F1-score, TP, and FP are shown. The mean values were determined using all 171

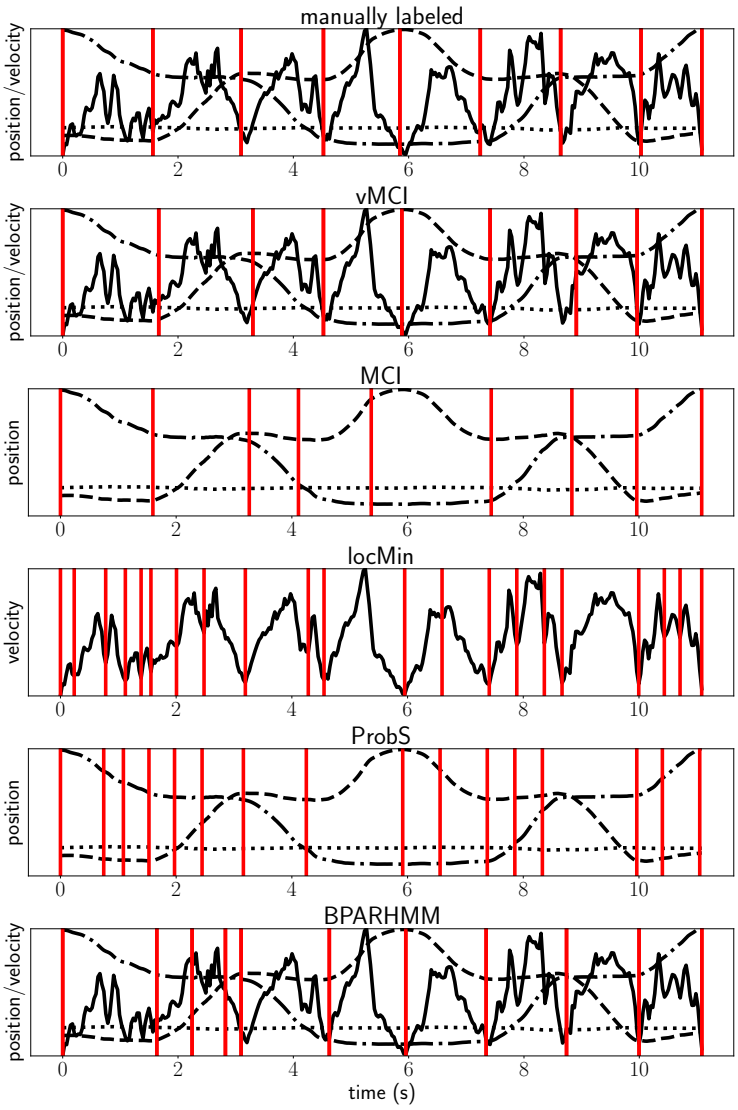


Figure 4.6.: Example demonstration of a movement through the step pattern, consisting of the building block sequence *right*, *up*, *right*, *up*, *down*, *left*, *down*, *left*. Shown is the absolute velocity (solid line) and the position (x,y,z - dashed/dotted lines) of the hand. The top left plot shows the manual segmentation result. The other plots show the result of vMCI, MCI, locMin, ProbS, and BPARHMM, in which the data which is used as basis for segmentation is plotted (position and/or velocity). *Images extracted from [Gutzeit and Kirchner, 2022], Figure 6.*

demonstrations of going through the step pattern upwards and back, resulting in an optimal segmentation into 8 building blocks which corresponds to an optimal number of true positives of 7. A determined segmentation point was treated as correct, when it lay within a margin around the ground truth segmentation point of 0.2 seconds. Using the simple locMin approach, all segmentation points were detected, but with a mean value of 3.7 FP. The lowest number of FP was achieved using vMCI, which detected on average 6 segmentation points with 1 FP. With this, the vMCI algorithm had the highest F1-score on this dataset and outperformed all other approaches, whereby locMin and BPARHMM over-segmented the data. ProbS was not able to detect the segmentation points at all, possibly because there was only little variation in movement execution in the data, i.e., nearly the whole demonstration can be represented by the same DMP. However, on movement examples with a noisy velocity, ProbS over-segmented the data.

The dataset contained examples with very smooth velocity changes as well as noisy velocity profiles. An example result of the segmentation using the different approaches on a sample with noisy velocity is shown in Figure 4.6. For this example, vMCI achieved a perfect segmentation despite the noise in the velocity. However, this noise resulted in an over-segmentation using locMin, ProbS, and BPARHMM.

4.4.4. Segmentation of stick-throwing movements

In the stick-throwing dataset, 34 throwing demonstrations were recorded. The datasets consists in the manually determined ground truth segmentation of 126 movement segments. On this dataset, the same parameters configurations were used as for the step data for vMCI, MCI, BPARHMM, and ProbS. The window for locMin was set to 0.2 seconds and 0.07 seconds for initial over-segmentation needed in ProbS.

The mean segmentation results for all algorithms can be seen in Table 4.3. For this dataset, the optimal number of TP is 3.7, i.e., the throwing movements were on average segmented into the three main building blocks *strike out*, *throw*, and *swing out*, and some occurrences of additional segments. Again, the highest number of detected TP was achieved using locMin and the lowest number of FP using vMCI, where both algorithms had a similar F1-score of approximately 0.7. With that, vMCI and locMin outperformed all other approaches. On this dataset, in which nearly no noise can be observed, BPARHMM and locMin again detected some FP, but not as much as on the

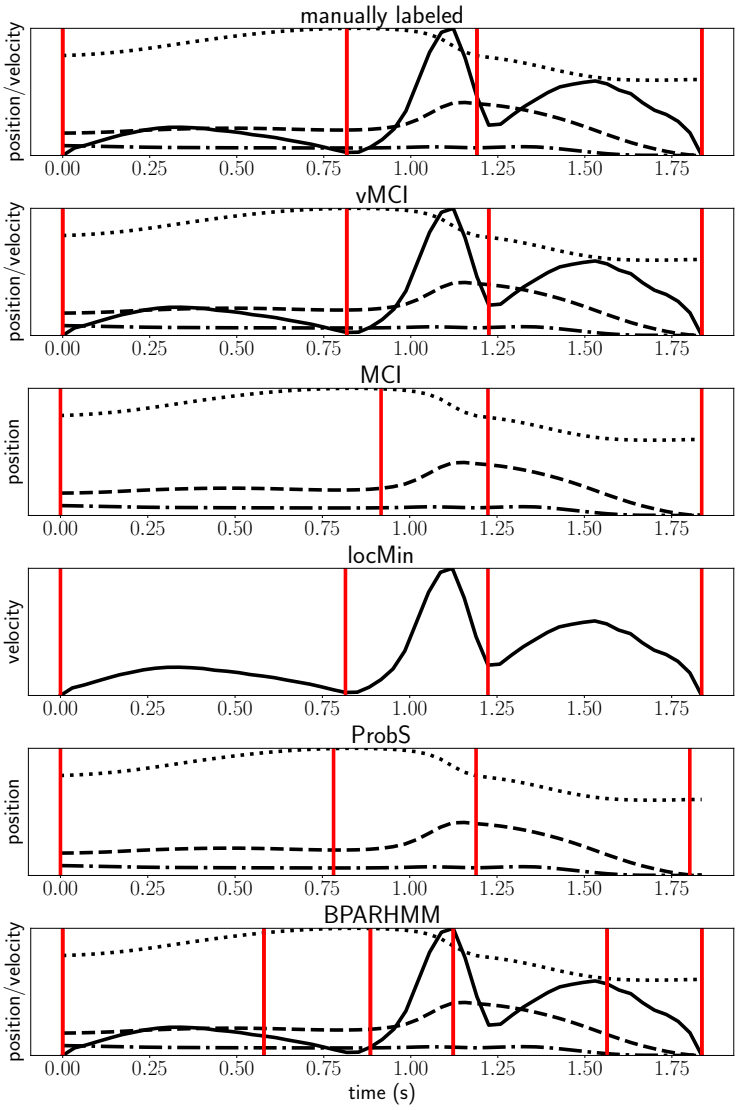


Figure 4.7.: Example demonstration of a stick-throwing movement, consisting of the building blocks *strike out*, *throw*, and *swing out*. Shown is the absolute velocity (solid line) and the position (x,y,z - dashed/dotted lines) of the hand. The top left plot shows the manual segmentation result. The other plots show the result of vMCI, MCI, locMin, ProbS, and BPARHMM, in which the data which is used as basis for segmentation is plotted (position and/or velocity). *Images extracted from [Gutzeit and Kirchner, 2022], Figure 7.*

Table 4.3.: Mean segmentation results on stick-throwing movements.

	F1-score	num. TP (opt.: 3.7)	num. FP (opt.: 0)
vMCI	**0.70**	1.9	**0.9**
MCI	0.58	1.5	**0.9**
locMin	**0.71**	**2.0**	1.8
BPARHMM	0.56	2.0	2.5
ProbS	0.23	0.6	1.0

step data. Again, ProbS did not perform well on this data. An example result can be seen in Figure 4.7.

4.4.5. Segmentation of pick-and-place movements

To segment the pick-and-place dataset, the recorded data was down-sampled to 20 *Hz*, because in this dataset the movements were on average much slower (mean segment length of (1.3) seconds) compared to step dataset (mean segment length of 0.82 seconds). The dataset contains in the manually determined ground truth 172 movement segments. The parameters for vMCI, MCI, BPARHMM, and ProbS were identical to the ones in the previous evaluations. The range in which the minima were detected in locMin was set to 0.2 seconds and 0.1 seconds for the initial over-segmentation in ProbS. Due to the slower movement velocity, the margin in which a detected segmentation point was still treated as correct was increased to 0.3 seconds.

This dataset has a bigger movement variability compared to the other two datasets and the ground truth segmentation was more difficult to manually define. Due to rather slow movements when the object was grasped or placed, the exact movement start and end point were difficult to determine based on the position of the hand. If the subject did not move or did a movement which was not part of the defined movement classes, the segment borders were still added to the ground truth data. To accomplish the pick-and-place tasks, the subject had to perform 6 basic movements: *approach forward*, *move object to table*, *move to rest right*, *approach right*, *move object to shelf*, *move to rest down*. In between there were *idle* phases. The average number of manually defined segmentation points per demonstration was 8.1.

As can be seen in the results in Table 4.4, most TP were detected using BPARHMM

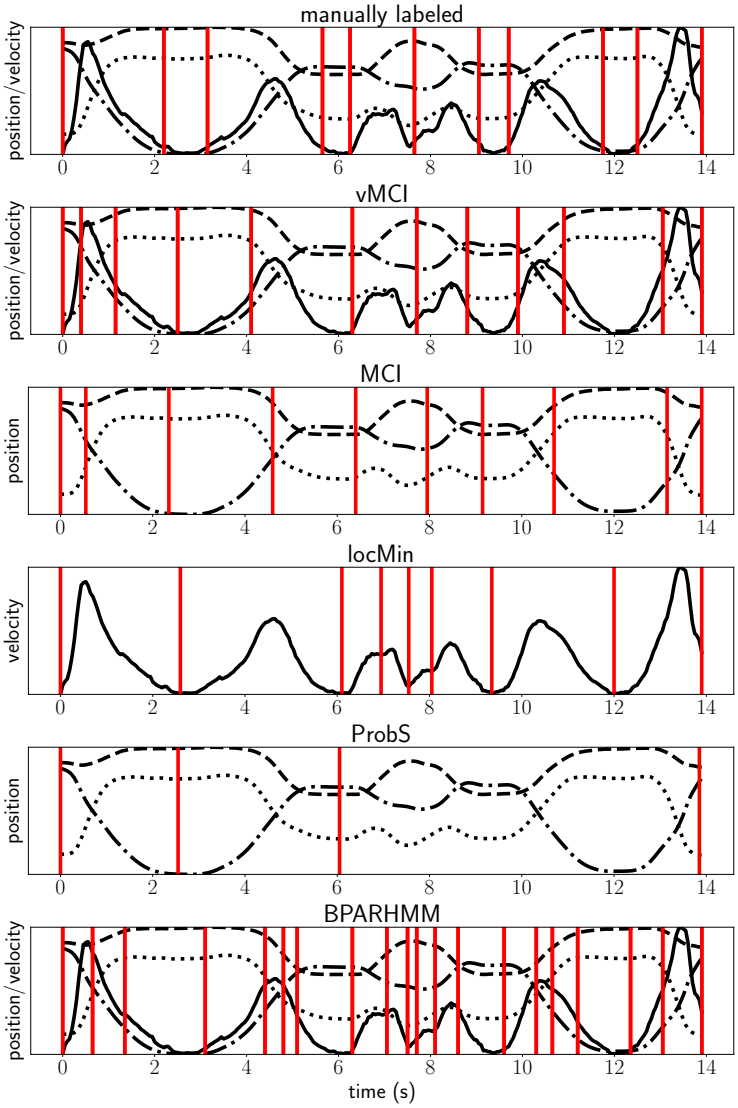


Figure 4.8.: Example demonstration of the pick-and-place movement, consisting of the building blocks *approach forward*, *move object table*, *go to rest right*, *approach right*, *move object shelf*, *move to rest*, with small periods of no movement in between (building block *idle*). Shown is the absolute velocity (solid line) and the position (x,y,z - dashed/dotted lines) of the hand. The top left plot shows the manual segmentation result. The other plots show the result of vMCI, MCI, locMin, ProbS, and BPARHMM, in which the data which is used as basis for segmentation is plotted (position and/or velocity). *Images extracted from [Gutzeit and Kirchner, 2022], Figure 8.*

Table 4.4.: Mean segmentation results on pick-and-place movements.

	F1-score	num. TP (opt.: 8.1)	num. FP (opt.: 0)
vMCI	0.67	5.0	3
MCI	0.60	4.9	4.0
locMin	**0.79**	6.5	3.0
BPARHMM	0.52	**6.3**	10.7
ProbS	0.32	1.6	**1.1**

and locMin, but with a high number of FP using BPARHMM. On this data, vMCI resulted in a higher number of FP compared to the other two datasets. If the results on the individual demonstrations are examined more closely, one can observe that vMCI detected most of the segments but with a very inaccurate position of the segment boundaries, which lay often outside the margin of 0.3 seconds. This can also be seen in the example result in Figure 4.8. Again, the segmentation points could not be reliably detected using ProbS.

4.5. Discussion

In the comparison of several features in human point-to-point movements in section 4.2, a bell-shaped curve in the absolute velocity of the hand was identified as a feature which was identical for all movement segments, independent from the movement direction. This observation could also be verified on the more complex pick-and-place movements. In the experimental evaluations of vMCI run on the point-to-point movements of the step reference setup (section 4.4.3) it could be seen that vMCI was able to detect these segments with bell-shaped hand velocity. Also in movement examples with a very noisy velocity, the majority of the true segmentation points were detected using vMCI. Based on the velocity features, segmentation points could also reliably be detected using locMin. In comparison to vMCI, the resulting segment borders were more accurate, because they were located directly at time points where the velocity was in a local minimum. However, the threshold in locMin needed to be adapted to different datasets or even different movement examples of the same tasks executed in different velocities. Furthermore, in data with a noisy velocity profile

locMin over-segmented the data and detected a lot of false positives. By using vMCI this over-segmentation can be prevented without an additional preprocessing, such as smoothing. The performed experiments show that the algorithm is robust against noise and can handle variations in the movement execution more effectively than other methods for unsupervised segmentation. Furthermore, the influence of the parameters can be reduced compared to MCI which makes the algorithm applicable to different datasets where the required manual user input is reduced.

In the evaluation performed on the stick-throwing and pick-and-place data, vMCI again yielded good results and detected the majority of the segmentation points correctly. The ground truth segmentation points were determined based on the recorded marker positions and the vMCI approach detected these based on the position and velocity of the hand marker. This supports the assumption that in these more complex manipulation movements compared to the point-to-point movements, building blocks can still be detected using a bell-shaped velocity of the hand.

In all analyzed movements, but especially in the pick-and-place data, vMCI detected some segment points inaccurately, resulting in a lower number of TP with increasing number of FP. However, the total number of detected segments was nearly the same as the manually defined number of segments in all experiments. This indicates that the data is not over-segmented using vMCI.

On all three datasets, vMCI performed better than the MCI algorithm which does not model the velocity dimension of the data differently than the position. This also applies to the comparison to BPRAHMM, which models the data in a very similar way as vMCI and MCI but does not have the special look at the velocity. Using BPARHMM, also good results could be achieved but the algorithm was more sensitive to noise. Furthermore, it is a batch method which needs a lot more computation time and cannot be run online. Possibly, this results from the design of the algorithm which identifies a set of basic movement the analysed demonstrations are generated from. In the experiments of Lioutikov et al, the algorithm was tested on movement demonstrations which contain different combinations of a small number of basic movements. In that data, ProbS successfully identified basic movements [Lioutikov et al., 2017]. Our datasets, on the other hand, contain repetitive demonstrations of the same manipulation movement without changes in the order of the concatenated building blocks. Our results indicate, that ProbS is not suited to segment this kind of data. In the performed experiments, it was not possible to achieve good results using

the reimplementation of the ProbS algorithm. Although it was shown in [Lioutikov et al., 2017] that the algorithm converges, this could not be achieved in a reasonable time period in the evaluations presented in this chapter and calculations had to be stopped after a fixed number of iterations. On all observed movements, only unsatisfying results could be obtained using this method. However, BPARHMM and ProbS give not only the segmentation points but also clustered segments that describe which segments belong to the same movement. For vMCI, a successive step is needed to obtain grouped or labeled segments. Due to the simple structure of the building blocks detected using vMCI, it will be shown in chapter 5 that labeled segments can be obtained using a simple k-Nearest Neighbor classifier with $k = 1$ and a small number of training examples.

5 Chapter 5. Few-shot Recognition of Building Blocks

In this chapter, methods to recognize building blocks in human manipulation movements using a small number of training examples are compared. Repetitions of the same movement building blocks detected using the vMCI algorithm introduced in chapter 4 should be recognized, i.e., they should be annotated, to improve usability in applications such as imitation learning. By assigning suitable annotations to the identified movement segments, the selection of the behavior of interest becomes intuitive and easy to use in different interaction scenarios.

It is proposed in this chapter to use k-NN to classify movement building blocks detected using vMCI, as depicted in Figure 5.1. This simple approach has only one parameter, k, to tune. It is assumed that by choosing $k = 1$ in classification scenarios where segments obtained by vMCI should be automatically annotated based on a small training set size, reliable classification accuracies can be achieved. To verify this assumption, several experiments were performed on datasets of different complexity with a limited maximal number of examples per class in the training data. For this, lever-pulling, pick-and-place, ball-throwing, and stick throwing movements recorded using the Qualisys motion tracking as well as gestures recorded using the Xsens motion suit were used. An overview of the datasets is given in table 3.1, with detailed explanations about the data recording procedure given in chapter 3. In the presented evaluations, the stick-throwing data without tracking of the stick (stick-throwing 1 in table 3.1) which contains a higher number of movement examples than the other

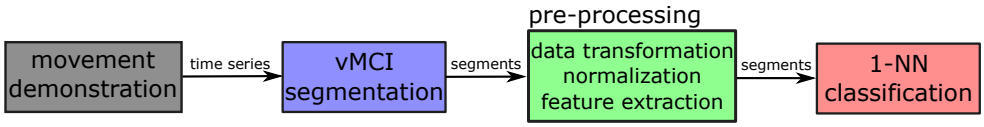


Figure 5.1.: Data processing chain to detect and recognize movement building blocks in human movement demonstrations.

stick-throwing set is used. The four manipulation movement datasets which were recorded using the Qualisys system were segmented into building blocks using the vMCI algorithm. The gesture data, which contains more complex movements that are not segmented into building blocks, was used as an additional dataset to evaluate the limits of simple 1-Nearest Neighbor (1-NN) classification.

A schematic overview of the performed experiments is given in Figure 5.2. In the first experiment, the 1-NN classification is compared to a classification using HMMs, which are widely used in the literature to recognize human movements, see section 5.1. This evaluation is done without hyper-parameter tuning on lever-pulling, pick-and-place, and ball-throwing movements to get a first impression of the performance of the algorithm on datasets of different complexity. In the second experiment, the first evaluation is extended with a hyper-parameter tuning of the classification approaches. In this experiment, k-NN and HMMs are compared to LSTM based-classification on stick-throwing demonstrations and gestures. These datasets are used for the hyper-parameter evaluation because they contain a higher number of movement examples and thus give a better representation of possible movement variations. In the third experiment, the same datasets were used to evaluate the generalization of k-NN, HMM, and LSTM to the movements of subjects whose demonstrations were not part of the training data.

This chapter is structured as follows: In section 5.1 related work is described and the compared algorithms are presented in section 5.2. The proposed data preprocessing and feature extraction mechanisms, including a discussion about the complexity of the datasets, is presented in section 5.3. The three conducted experiments and the results are presented in section 5.4. At the end of this chapter the results are discussed.

Most of the texts, figures, and tables in this chapter are taken or adapted from [Gutzeit and Kirchner, 2016], [Gutzeit et al., 2016], and [Gutzeit, 2021]. First preliminary work regarding online classification of movements using HMMs is presented in [Schreiter et al., 2014] and [Schreiter et al., 2015].

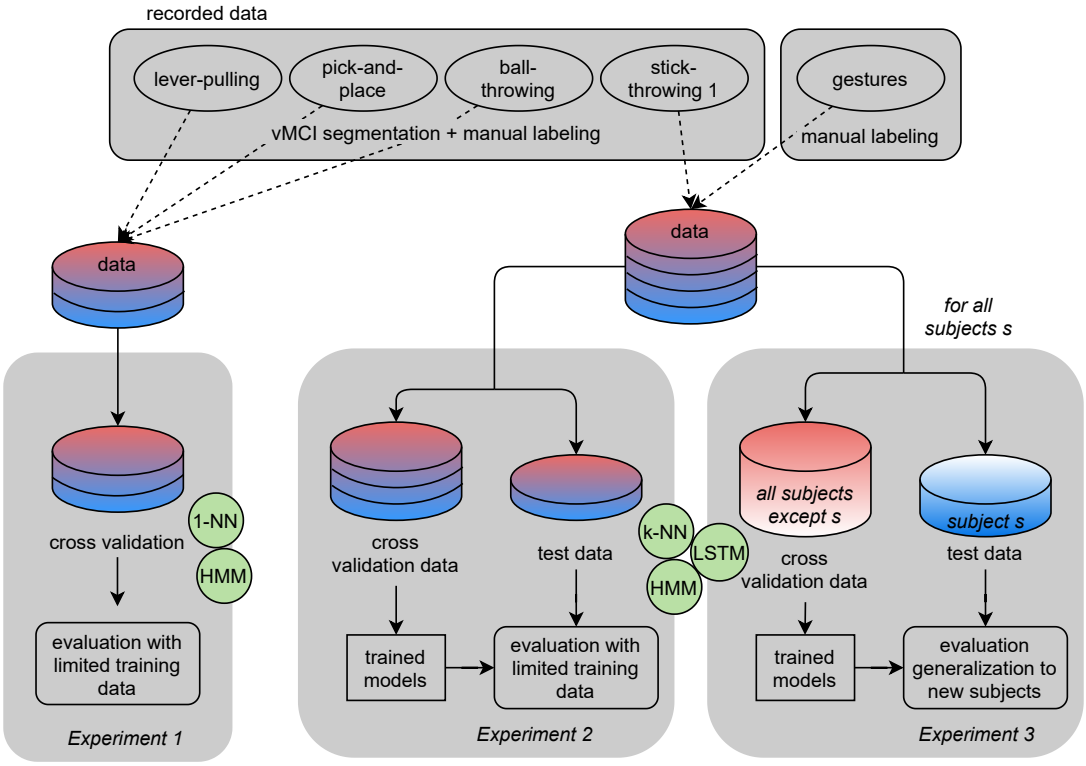


Figure 5.2.: Schematic overview of the approach to evaluate different classification methods using small training data set sizes and five different datasets. Details about the experiments are described in section 5.4. *Parts of the diagram are taken from [Gutzeit, 2021], Figure 1.*

5.1. Related Work

Human action recognition is an active research area with a lot of different applications and methods. In this section, an overview of the most important and widely used methods is given. Most approaches in the literature are based on the analysis of video or RGB-D data in applications such as the detection of tackles in soccer games, support of elderly in their homes, or gesture recognition in video games [Poppe, 2010]. In these approaches, large efforts have to be put into the detection of the human and its posture in the measured data streams. Afterwards, the observed actions are classified with algorithms such as Support Vector Machines, or their probabilistic variant the Relevance Vector Machines, HMMs, k-NN or neural network-based approaches, see [Poppe, 2010] and [Zhang et al., 2019] for a detailed overview.

In the last decades, HMMs were widely used to classify human actions and gestures. For example, in [Stefanov et al., 2010] and [Aarno and Kragic, 2008], HMMs were used to recognize human intentions in teleoperation scenarios. Borghi et al. proposed an online double-stage Multiple Stream Discrete HMM to classify gestures from 3D joint positions acquired with a Kinect [Borghi et al., 2016]. With this approach, high classification accuracies could be achieved on three public and a new recorded data set containing different actions created for human computer interaction. On the other hand, Corbonera Luvizon et al. performed an extensive preprocessing to extract sets of spatial and temporal local features of skeleton data recorded with a Kinect and combined them using a metric learning approach [Carbonera Luvizon et al., 2017]. With this method the classification accuracy of a k-NN classifier could be improved on three different public data sets.

Recently, neural network-based approaches became popular in all pattern recognition domains. Patsadu et al. compared a neural network with a Support Vector Machine, a decision tree, and Naive Bayes to distinguish the movement patterns *stand*, *sit down*, and *lie down* recorded with a Kinect camera [Patsadu et al., 2012]. In the huge data set with more than 10.000 recordings, the best performance was reached with the neural network approach. Long-term motions in video sequences were detected in [Shi et al., 2017] using a method based on a Convolutional Neural Network (CNN)-Recurrent Neural Network (RNN) network. To handle unreliable data, Liu et al. introduced a new gating algorithm for LSTMs [Liu et al., 2017]. Spatial and temporal dependencies between joints are learned to recognize human actions in skeleton data. Unreliable data, which can result from noisy data or occlusions, are handled with a newly introduced trust gate added to the LSTM.

An approach which combines movement segmentation and clustering in time series data is presented in [Fod et al., 2002]. In their work, human arm movements were segmented into so-called movement primitives at time points where the angular velocity of a certain number of degrees of freedom crosses zero. After a Principal Component Analysis (PCA)-based dimensionality reduction, the detected movements were clustered using k-Means. Gong et al., on the other hand, proposed Kernelized Temporal Cut to segment full body motions, which is based on Hilbert space embedding of distributions [Gong et al., 2014]. In their work, different actions were recognized using Dynamic Manifold Warping as similarity measure.

However, most of the approaches in the literature were applied to precisely specified

movements. The performance with respect to naturally and intuitively performed movements has not been analyzed. Furthermore, many approaches rely on huge sets of labeled data. If these are not available for a certain application, the training datasets have to be manually generated, which requires a large human effort. To reduce this effort, algorithms which give reliable results on small dataset sizes are beneficial. This new research area is known as Few-shot learning, a survey is presented in [Wang et al., 2020]. Usually, few-shot learning algorithms combine limited labeled training examples with pre-trained models which contain prior knowledge of the tasks, e.g., by including labeled data from other domains [Wang et al., 2020]. In contrast to this, standard machine learning approaches are evaluated in this chapter with respect to their performance on classifying manipulation movements of different complexity which were segmented into building blocks using limited training data.

5.2. Evaluated Recognition Algorithms

In this chapter, the performance of k-NN to recognize human movements recorded using motion tracking is evaluated. It is compared to widely used recognition methods based on HMMs and LSTMs. The usage of these three algorithms for movement classification is briefly described in this section.

5.2.1. k-Nearest Neighbor

In the k-NN classification, an observed movement sequence is assigned to the movement class, which is the most common among its k closest neighbors of the training examples. To determine the closest neighbors, the standard Euclidean distance metric is used in this thesis. For this, all segments are interpolated to the same segment length. Alternatively, Dynamic Time Warping (DTW) could be used as a distance measure. However, in a preliminary analysis of k-NN classification on manipulation behaviors the presented approach outperformed a DTW-based k-NN. The k-NN algorithm does not need much parameter tuning, as it has just one hyper-parameter k. To classify the recorded data sequences with k-NN, the feature trajectories for each movement recording are transformed into a single feature vector.

5.2.2. Hidden Markov Model

HMMs are very common probabilistic models for time series data. In the model it is assumed that a data sequence is generated by a Markov process with states that are not directly observable. The observable output is influenced by the transition probabilities between states and their emission probabilities. A detailed introduction is, e.g., given in [Bishop, 2006]. In the experiments in this chapter, one HMM with Gaussian emissions is trained for each class in the data using the Baum-Welch algorithm. A new data sample is assigned to the label of the HMM from which it is most likely generated. For each HMM, the number of hidden states h must be set.

5.2.3. Long Short-Term Memory Network

LSTMs are artificial RNNs especially designed to process time series data. They were firstly presented in [Hochreiter and Schmidhuber, 1997]. Standard RNNs connect the output of previous time steps to the current input in order that sequences of data can be processed. In extension to this, a LSTM contains an input gate, an output gate and a forget gate which regulate which information goes into and out of the LSTM cell. With this, data points over long time intervals can be saved and weighted so that information that is no longer important can be forgotten.

In this thesis, a simple structure with one LSTM layer is used. The input layer contains one neuron for each feature, which is fully connected to the LSTM layer. As output, a Dense layer with softmax activation function is used, which has a single neuron for each class. During training, the categorical cross entropy is used as error function. To prevent over-fitting, early stopping is applied and training is stopped if the accuracy on a validation dataset did not increase in the last p epochs, where p is called patience value. For this architecture, different numbers of cells c, different batch sizes b, and patience values p are compared.

5.3. Data Preprocessing and Feature Extraction

Before classification, the motion tracking data needs to be preprocessed. The applied data processing chain is depicted in Figure 5.1 and is described in detail in this section.

5.3.1. Preprocessing

The recorded data trajectories contain several repetitions of the performed task and need to be segmented into individual movements to evaluate the classification approaches. For some datasets (see table 3.1) this was done manually using the trajectory visualization tool as described in section 3.8. Because this is very time consuming, especially for datasets with a high number of recorded movements, some of the datasets were segmented automatically using the vMCI algorithm. To create datasets for evaluation, a reliable ground truth should be generated in which each segment is verified by a human. For this reason, all detected segments were manually labeled. However, some of the automatically obtained segments could not be assigned to one of the expected classes because the movement was only partly covered. This could result from errors in the segmentation as well as from demonstrations where a movement was slowed down before the movement class ends. A case would be when the subject thought about the exact position to grasp the object and thus slows movement speed before reaching the object. An example can be seen in Figure 5.4b. The concatenation of the first two detected segments belong to the class *approach forward*. Nonetheless, the vMCI algorithm detected two segments, both with a bell-shaped velocity curve, because the subject slowed down the movement right before reaching the object. These incomplete movement segments were discarded for the evaluation of the classification approaches. Furthermore, some of the detected movement segments did not belong to one of the predefined movement classes of the experiment. Usually, these non-assignable segments belonged to small extra movements, that were not part of the main movement task and thus were not considered in the defined movement classes. These movement segments were also excluded from the evaluation of the classification.

The segmented time series contain raw marker positions recorded in Cartesian coordinates in the global coordinate system of the motion capture system. This results in different time series if the same movement is executed at a different position. Thus, the data is transposed into a coordinate system which is not global but relative to the human demonstrator. For movements recorded with the Qualisys system the coordinate system spanned by the marker cluster placed on the back of the subject (see Figure 3.1) is used as reference point. For Xsens recordings the reference system is defined by the anatomical landmarks `RightShoulder`, `LeftShoulder`, and `T8`, which

is a position at the spinal cord of the internal model of the human as described in the user manual of the AVN Avinda sensor suit[2]. Each data point of a segment is transformed into this reference coordinate frame recorded at the first time point of this segment.

5.3.2. Feature extraction

For each recorded movement segment, different features can be calculated. Depending on the tracking system, these are directly measured or can be calculated easily from the raw data. The following features were determined from the recorded data:

1. transformed, human-relative 3D positions of the markers placed on the body of the human,
2. absolute velocity of these marker positions,
3. orientation of the hand,
4. elbow joint position determined by calculating the angle between lower and upper arm,
5. elbow joint velocity calculated using the temporal derivative of the elbow joint position,
6. shoulder joint position determined by calculating the angle between upper arm and upper body,
7. shoulder joint velocity calculated using the temporal derivative of the shoulder joint position,
8. distance between hand and object,
9. object velocity.

To compare 1-NN to HMMs on the lever-pulling, pick-and-place, and ball-throwing data, only the transformed positions of the hand, elbow, and shoulder marker were used as well as the distance of the human hand to the manipulated object and the object velocity for the pick-and-place movements. All segments were interpolated to the mean segment length of the current data set using Spline interpolation. In

the other experiments, k-NN, HMMs, and LSTMs were evaluated on gesture and stick-throwing data using features 1-7 with an interpolation to a length of 25 time points using spline interpolation. Since the range of the individual features varies, all features were normalized to values in the range [0, 1].

5.3.3. Data complexity

The five datasets used for the evaluation in this chapter contain data of different complexity. The lever-pulling movements are rather simple because the movement direction is predetermined by the lever during pulling which only allows for little variations between movement executions. In contrast to this, the pick-and-place data show more variations because the subjects moved their arms free towards the box and the table. However, the box was always grasped from or placed to approximately the same position and only 26 movement examples were recorded. For the ball-throwing data, more movement demonstrations of more subjects were recorded. Although the throwing movements were restricted, because the ball had to be thrown always "from top", higher inter-subject variability can be observed in the data due to different throwing skills of the subjects.

The gesture and stick-throwing data were compared in more detail by transferring the determined features of all recordings into a two-dimensional manifold using t-distributed Stochastic Neighbor Embedding (t-SNE) [van der Maaten and Hinton, 2008]. The result, in which samples with a low feature distance are close, is shown in Figure 5.3. The manifold transformation of the gesture data can be seen in Figure 5.3a. Although clusters for the different movement classes can be observed, the clusters of the 11 classes clearly overlap, i.e., the classes are not clearly separated. On the other hand, movement trajectories of different subjects of the same gesture can be separated in this visualization, as the subject samples show clusters within one gesture class. Thus, the generalization to new subjects is a challenging task for this heterogeneous dataset.

The movement classes of stick-throwing data are separated more clearly, see Figure 5.3b. Although samples of the same class performed by different subjects can also be distinguished in this data, the distances to the other classes are larger compared to the gesture data. Only the two classes *idle* and *strike out* overlap in the manifold. This shows the much lower complexity of this data compared to the gesture data. This

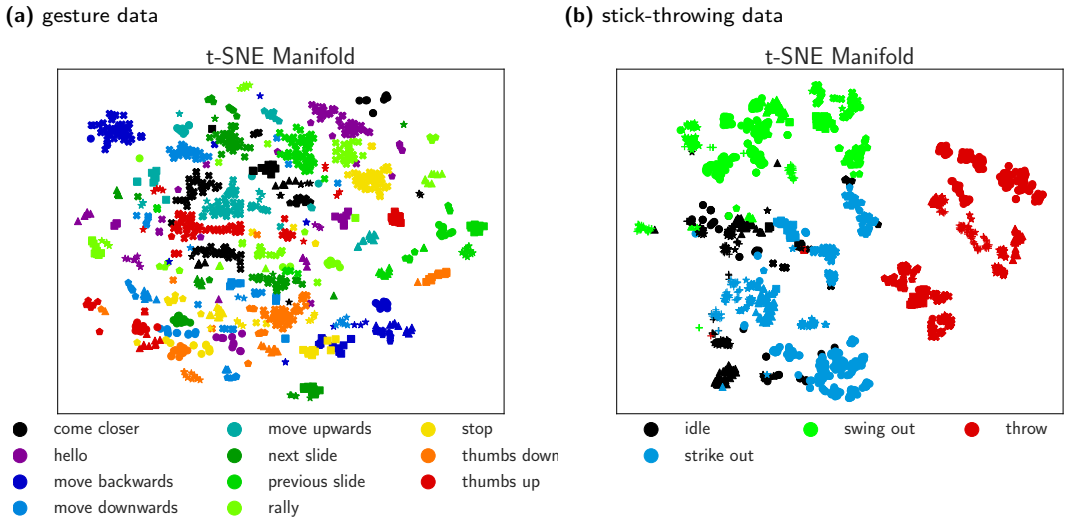


Figure 5.3.: T-SNE manifolds of the gesture data (a) and the stick-throwing data (b). Each movement class can be identified by a different color, samples of the different subjects have different markers. *Images extracted from [Gutzeit, 2021], Figure 4.*

has several reasons. First, the gesture data contains more classes and some of them are very similar in their execution. For example, the movement classes *thumbs up* and *thumbs down* differ only in the orientation of the hand. In the stick-throwing dataset the task is to throw a stick to a certain position. This is in contrast to the movements in the gesture data a goal-directed behavior, in which less variations can be assumed. Furthermore, the stick-throwing data was segmented into its main movement blocks characterized by a bell-shaped velocity profile using the vMCI algorithm which further reduces complexity whereas the gesture data contains gestures which consist of several building blocks. For example, the gesture *hello* contains several repetitions of waving the hand and their concatenation forms one example of the movement class (a detailed description of the performed gestures is given in section 3.6). That means, one movement class in the gesture dataset can contain a concatenation of bell-shaped velocity profiles whereas the stick-throwing movement classes always have a single bell-shaped velocity per movement example. This results, next to the different number of movement classes in the two datasets, in a lower complexity of the stick-throwing data compared to the gesture data.

	movement classes	num. examples
lever-pulling	move lever	62
	approach forward	76
	move to rest	72
	idle	72
pick-and-place	approach forward	20
	move obj table	26
	move to rest right	25
	approach right	23
	move obj shelf	26
	move to rest down	24
	idle	11
ball-throwing	strike out	221
	throw	227
	swing out	339
	idle	208

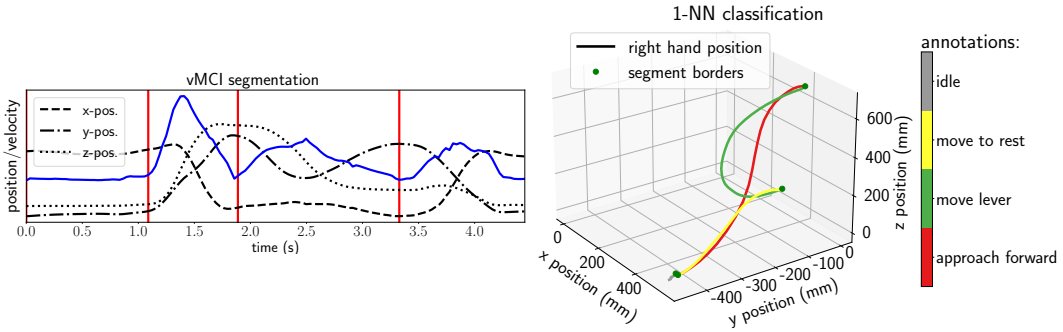
Table 5.1.: Occurrences of each class in the lever-pulling, pick-and-place, and ball-throwing data.

5.4. Experiments and Results

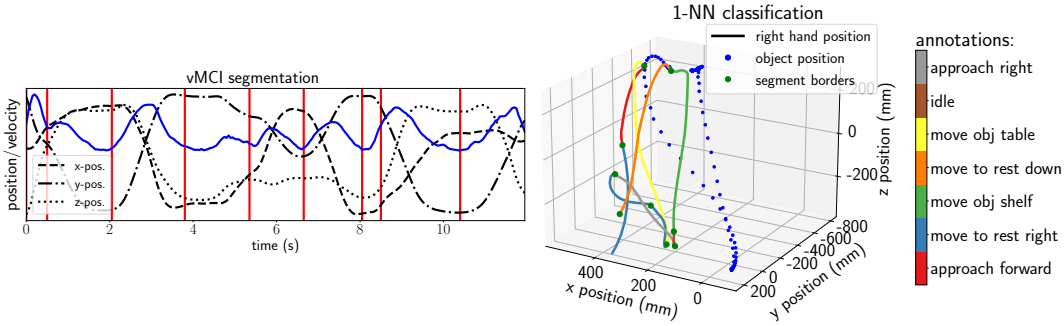
5.4.1. Experiment 1: Evaluation of 1-NN in comparison to HMMs

1-NN and HMMs based classification were compared using a stratified cross-validation with a fixed number of examples per class in the training data on the lever-pulling, pick-and-place, and ball-throwing data. The training set sizes were varied from 1 example per class to 20 randomly selected examples per class and the remaining data was used for testing. Since the performance of the classification with small training set sizes should be evaluated, the maximal number of training examples per class was kept low. For each number of examples per class in the training data, the cross-validation was performed with 100 iterations. The number of states in the HMMs was determined with a stratified 2-fold cross-validation repeated 50 times with equally sized training and validation sets. As a result, each HMM was trained with one hidden state.

(a) lever-pulling example



(b) pick-and-place example



(c) ball-throwing example

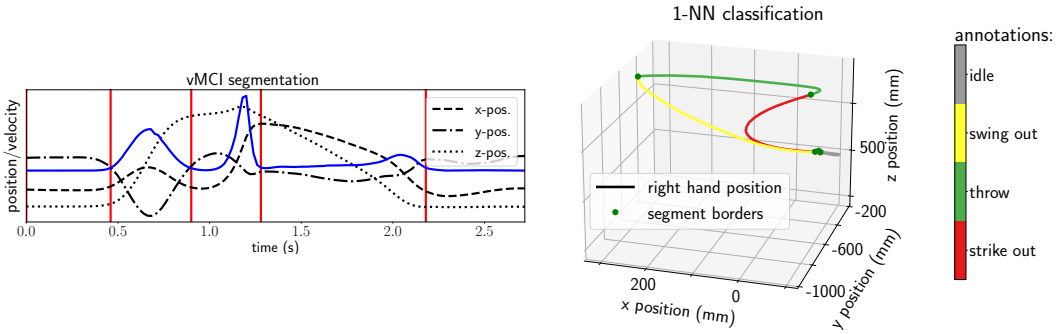


Figure 5.4.: Example segmentation and classification results of (a) lever-pulling, (b) pick-and-place, and (c) ball-throwing movements. On the left side, the segmentation results are shown. The x-, y-, and z-position of the hand are visualized with black lines. The blue line corresponds to its velocity and the red vertical lines are the segment borders determined by the vMCI algorithm. On the right side, the classification results of the same movement demonstrations are shown. The data was classified separately for each datasets using 1-NN, with only one example per class in the training data for the lever-pulling demonstration, and five examples per class in the training data for the classification of the pick-and-place and ball-throwing movement examples.

Segmentation and recognition of lever-pulling movements Although the demonstrations of the lever-pulling movements showed not always smooth bell-shaped curves, the vMCI algorithm successfully segmented the trajectories without any adaptations of hyper-parameters or an additional preprocessing of the data. An example of the segmentation results can be seen in Figure 5.4a. The resulting movement segments were manually labeled into one of the 4 movement classes *approach forwards*, *move lever*, *move to rest*, and *idle*, that are present in the lever-pulling task. The occurrences of each class can be found in Table 5.1. Figure 5.4a shows the classification result of the movement example using 1-NN and only one example per class in the training data. The four movement segments were correctly classified into one of the predefined classes.

The results of the cross-validation comparing 1-NN with HMM-based classification are visualized in Figure 5.5a. 1-NN outperforms HMM-based classification in the case of small training set sizes, especially for a very small (<5) number of examples per class in the training data. Indeed, very high accuracy can already be achieved with one training example for each class with a mean accuracy of 95.3% using 1-NN but only 42.1% by using HMM-based classification. Using 1-NN, a mean classification accuracy of 99.0% is accomplished using 4 examples per class during training. In contrast to this, the same accuracy is not reached using HMM-based classification in this evaluation.

Segmentation and recognition of pick-and-place movements The demonstrations of the pick-and-place task could be successfully segmented into movement parts with a bell-shaped velocity profile using the vMCI algorithm. Two examples of the segmentation results can be seen in Figure 5.4b. The resulting movement segments were manually labeled into one of the 7 movement classes described in section 3.8. This resulted in 155 labeled movement segments with different occurrences of each class, as summarized in Table 5.1.

As described in Section 5.3, the distance of the hand to the object and the object velocity were calculated as additional features in this experiment next to the positions of the markers attached to the subject. An example result of the classification using 1-NN is shown in Figure 5.4b. For this example, demonstration of the pick-and-place task, all segments were labeled with the correct annotation using a training set with 5 examples for each class.

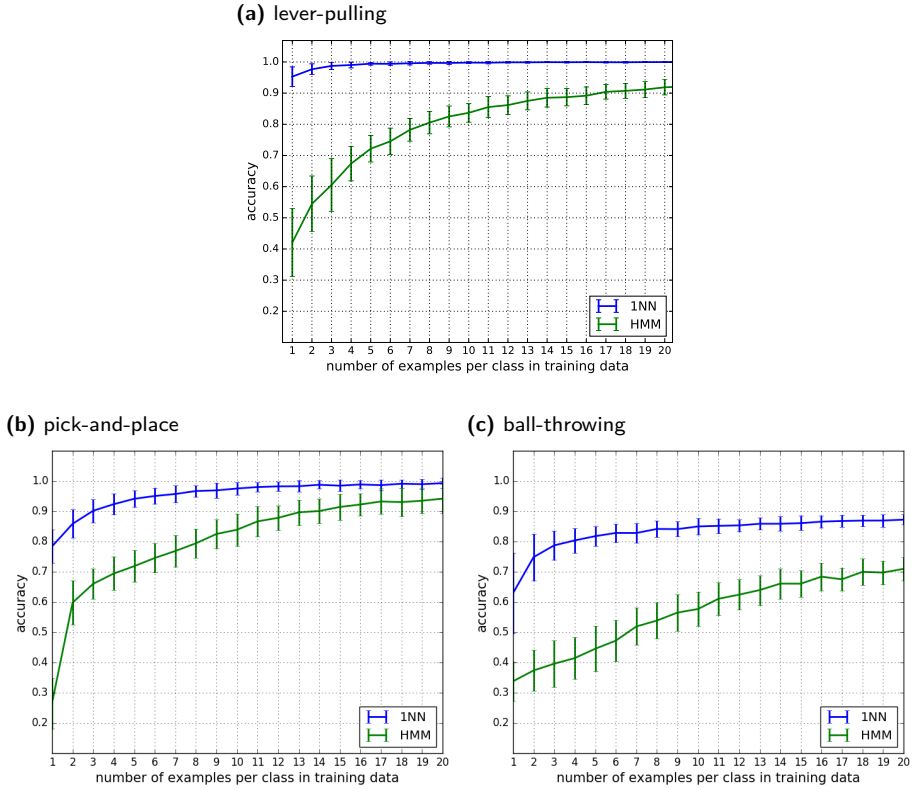


Figure 5.5.: Comparison of the accuracy of the classification of lever-pulling, pick-and-place, and ball-throwing movement segments using 1-NN and HMM-based classification. *Images extracted from [Gutzeit et al., 2016], Figure 6, 7 (c) and 8 (c).*

The results of the cross-validation using 1-NN and HMM-based classification are shown in Figure 5.5b. Because the data contains 7 different classes, an accuracy of about 14% can be achieved by guessing. Because the movement class *idle* contains only 11 movement examples, it was only part of the validation with maximal 10 examples per class in the training data. The 1-NN classification clearly outperforms the HMM-based classification using training sets with occurrences of each class smaller or equal to 20. Already with one example per class a mean accuracy of nearly 80% can be achieved using 1-NN. With 10 examples per class, the mean accuracy is 97.5% and with 20 examples per class 99.2%. In contrast, 14 examples per class are needed in the HMM-based classification to achieve an accuracy of 90% in this evaluation. With not more than 10 examples per class, the accuracy of the HMM-based classification is

considerably below the achieved accuracy using 1-NN.

Segmentation and recognition of ball-throwing movements Also in the ball-throwing data, meaningful segments could be detected using vMCI. A representative example of the segmentation result is shown in Figure 5.4c. Segment borders were correctly detected at positions were bell-shaped curves of the velocity profile end. The resulting segments of all 240 ball-throw demonstration were manually assigned to one of the four movement classes to evaluate the classification. Again, each class had a different number of occurrences in the available data, as summarized in Table 5.1. Figure 5.4c shows an example classification result using 1-NN and 5 examples per class in the training data. The 5 movement segments were correctly classified into one of the predefined classes.

The results of the cross-validation comparing 1-NN with HMM-based classification are visualized in Figure 5.5c. Like in the lever-pulling and pick-and-place experiments, 1-NN outperforms HMM-based classification in the case of small training set sizes. This experiment contains considerably more demonstrated movements and higher variability along demonstrations compared to the pick-and-place task. In here, the difference between classification algorithms is more contrasting. With one example per class in the training data, an accuracy of 62.9% using 1-NN can be achieved and only 33.8% by using HMM-based classification. This experiment contains 4 different classes, i.e., an accuracy of 25% can be achieved by guessing. Using 1-NN, a classification accuracy of 80% is accomplished using 4 examples per class during training. In contrast to this, the same accuracy is not reached using HMM-based classification in this evaluation. For comparison, the evaluation was additionally conducted using 100 examples per class during training. This resulted in a mean accuracy of 91.5% using 1-NN, and 77.8% using HMM-based classification. That means that even if more training data is available, the 1-NN classification outperforms the HMM-based approach.

5.4.2. Experiment 2: Evaluation of different hyper-parameters for k-NN, HMM, and LSTM

In this experiment, k-NN was compared to HMM and LSTM based classification with respect to classification accuracy and computation times. In extension to experiment

1, different hyper-parameter settings were evaluated. For this, the largest datasets that were acquired for this thesis were used: the stick-throwing data consisting of 697 movement samples from seven different subjects and the gesture data containing 1045 samples of 11 gestures performed by six subjects. Additionally, the stick-throwing dataset contains building block movements detected using vMCI and the gesture datasets contains movements which were not split into individual building blocks and thus are more complex. These datasets were selected to evaluate the limits of 1-NN classification.

On the two datasets, the classification accuracy on small training sizes was evaluated. For this, i samples of each class were randomly selected and used to train the classifiers. The remaining samples were used for testing. This was repeated 10 times for each $i \in 1, 2, ..., 10, 15, 20$. The final models were tested on a test set which was not part of the cross-validation data, consisting of 10% of the original dataset. The cross-validation was done for each classifier with different hyper-parameter values.

Recognition of stick-throwing movements The validations on the stick-throwing data were performed with hyper-parameters set to $k \in [1, 3, 5, 7, 10]$ for k-NN, number of hidden states $h \in [2, 5, 10, 15, 20]$ for HMM, and number of cells $c \in [2, 5, 10, 15, 25]$ for LSTM with batch size $b \in [8, 16, 32, 128]$ and patience value $p \in [5, 10, 15]$. With a maximum of 10 examples per class in the training data, the best result was achieved with the LSTM classifier with $c = 25$, $b = 16$, and $p = 10$, leading to a mean accuracy of 94%. k-NN with $k = 1$ reaches a mean accuracy of 88% and the HMM classifier had a mean accuracy of 70% with $h = 2$.

With these hyper-parameter settings, the classification accuracies reached with the number of examples per class in the training data limited to values between 1 and 20 are visualized in Figure 5.6a. LSTM and k-NN classification can deal well with very small training sets on this data. With these two classifiers, an accuracy above 80% was reached with only 3-4 examples per class in the training data. With more examples per class, only small improvements can be observed. In comparison, the HMM classifier needed a minimum of 15 examples per class to achieve the same result.

Recognition of gestures Because of the higher complexity of the gesture data, the validation on this dataset was performed with hyper-parameters set to $k \in [1, 3, 5, 7, 10, 15, 20]$ for k-NN, number of hidden states $h \in [5, 10, 15, 20, 25]$ for HMM,

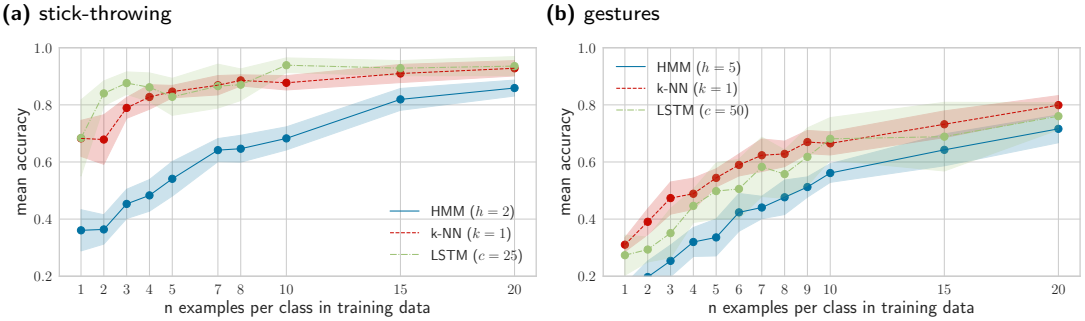


Figure 5.6.: Classification accuracy with best hyper-parameter setting of k-NN, HMM, and LSTM with limited number of training examples per class. The mean accuracies on the test data are shown as solid lines, with surrounding colored area highlighting the standard deviation. *Images extracted from [Gutzeit, 2021], Figure 6 (a) and 7 (a).*

and number of cells $c \in [5, 10, 15, 25, 30, 40, 50, 70]$ for LSTM with batch size $b \in [8, 16, 32, 128]$ and patience value $p \in [5, 10, 15]$. The results with limit $i = 10$ are shown in Figure 5.7. The hyper-parameters b and p of the LSTM classifier are fixed to $b = 16$ and $p = 10$, which gave the highest accuracies. With a maximum of 10 examples per class in the training data, the best result was achieved with the LSTM classifier with $c = 50$ cells, leading to a mean accuracy of 68%. k-NN with $k = 1$ had a similar mean accuracy (67%). HMM based classification did not achieve an accuracy above 60% with this small training size. For this dataset, the calculation times are also visualized in Figure 5.7. k-NN had the fastest calculation times because no model needs to be leaned and the method had short prediction times on the analyzed small datasets. The training time of the best LSTM network was around 1000 times slower, but after training the prediction time is similar to 1-NN classification. With HMM, training and prediction takes even longer.

In Figure 5.6b, the classification results with a number of examples per class in the training data between values from 1 to 20 are visualized. Hyper-parameters were set to $k = 1$ for k-NN, $c = 50$ for LSTM, and $h = 5$ for HMM classification. With these configurations, the highest accuracies could be achieved. The 1-NN classifier slightly outperformed LSTM classification in this experiment. With HMM classification accuracies dropped by $10 - 20\%$ in comparison to the other two methods. Especially with very small number of examples per class in the training set (≤ 10), HMM was clearly outperformed.

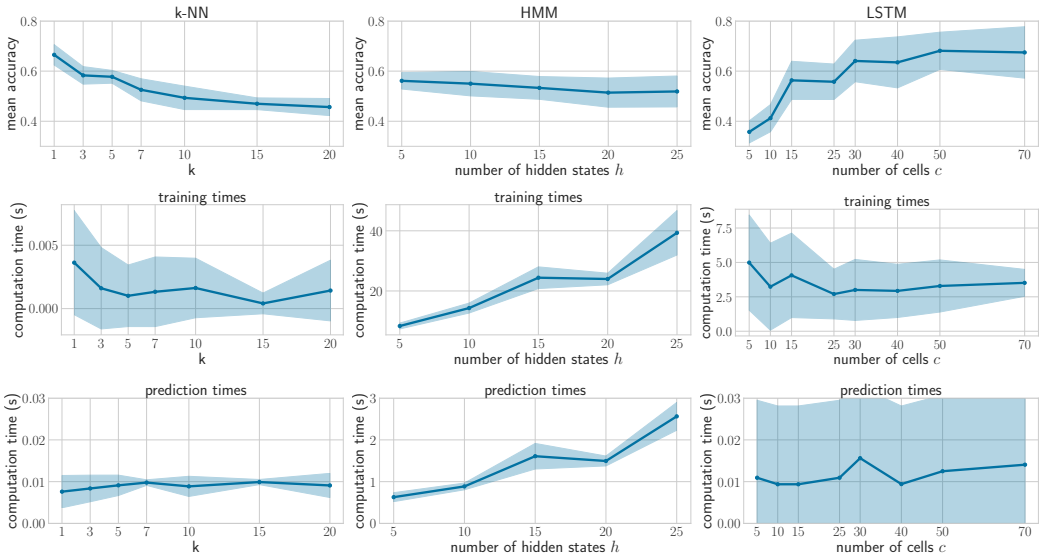


Figure 5.7.: Classification results for different hyper-parameter settings with limited training examples of the gesture data (experiment 2). Visualized is the classification with 10 examples per class. The top row shows the mean accuracy on the test data with different hyper-parameters of each classifier. Standard deviations are marked as colored areas. In the middle row, the training times are visualized, the bottom row shows the prediction times. All computations were run on a single core 3.7 GHz CPU without parallelization. Note the different axis scaling in the visualization of the computation times. *Images adapted from [Gutzeit, 2021], Figure 5.*

5.4.3. Experiment 3: Generalization to new subjects

In this experiment, the generalization to a new subject was evaluated on the stick-throwing and gesture data. For this, the k-NN, HMMs, and LSTMs were validated on the data of all subjects except one, using a limited training set containing i randomly selected examples of each class for each of the remaining subjects. In the validation data, the number of examples per class was fixed to 10 to avoid unbalanced classes. Final models were tested on the samples of the excluded subject which movements were left out for training. This was repeated 10 times for each subject and each limit i.

Stick-throwing data The results of the generalization capabilities of the classifiers on the stick-throwing data are shown in Figure 5.8a. In this evaluation, the hyper-parameter were set to $k \in [1, 3, 5, 10]$ for k-NN, $h \in [5, 10, 20]$ for HMM, and $c \in$

(a) stick-throwing **(b)** gesture

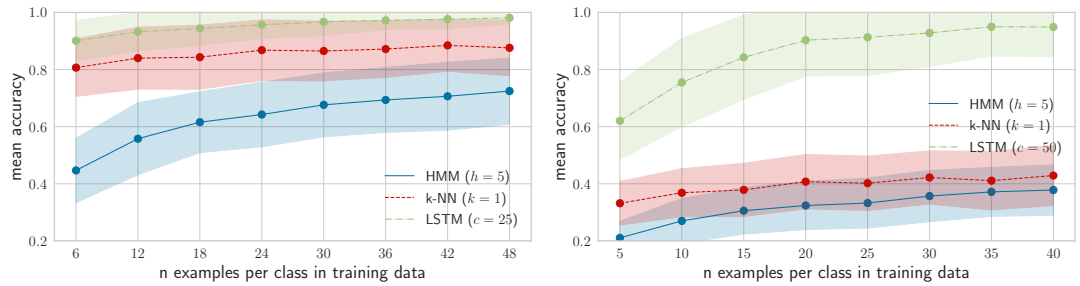


Figure 5.8.: Results of the leave-one-subject-out cross-validation comparing k-NN, HMM, and LSTM with limited number of training examples per class. The mean accuracy values are shown as solid lines, with surrounding colored area highlighting the standard deviation. *Images extracted from [Gutzeit, 2021], Figure 6 (b) and 7 (b).*

$[5, 10, 25, 40, 50]$ with $b = 16$ and $p = 10$ for LSTM. The best results were achieved with the LSTM classifier which generalized to new subjects with a mean accuracy above 80%, also with just 6 examples per class in the training data. The k-NN classifier also reached good accuracies despite the simpler model structure compared to LSTMs. The k-NN results are below the results of LSTM but still above 80%. A clearer drop in the accuracy can be observed in the HMM classification, with mean results below 80% for all training set sizes.

Gesture data The generalization to different subjects on the gesture data was evaluated with the same hyper-parameter settings as for the experiments with the stick-throwing dataset. The best results achieved with $k = 1$ for k-NN, $h = 5$ for HMM, and $c = 25$ for LSTM are shown in Figure 5.8b. The LSTM network was the only approach that classified the samples of subjects which are not part of the training data at a high mean accuracy around 90% if more than 4 examples of each class and each subject are used for training on this dataset. The mean accuracy of 1-NN and HMM were below 50% in this experiment. That means, on the gesture data, in which the movement examples do not correspond to building blocks movements, and which contains more variety in the movement demonstration between subjects, the simple k-NN model does cannot classify the movement data reliably in this experiment. This is discussed in more detail in the next section.

5.5. Discussion

In this chapter, it was proposed to use simple 1-NN classification with Euclidean distance measure to recognize human manipulation building blocks using a small number of training examples. This approach was evaluated in three different experiments on motion tracking data of different complexity. In the first experiment, 1-NN was compared to HMM based classification on lever-pulling, pick-and-place, and ball-throwing movements, which were segmented into building blocks using vMCI during preprocessing. All these manipulation movements of different complexity could be successfully classified with very limited training data. Although a supervised classification method like 1-NN always needs manually labeled training data, it was shown that the recognition of the movements can be done using a small set of training data, which considerably minimizes manual efforts. For the lever-pulling task, which is the simplest of the considered movements, a high classification accuracy of 95% could be achieved with just one training example per class. In comparison to widely used HMM-based movement classification, the accuracy was considerably higher using 1-NN on the three datasets of experiment 1.

In the two other experiments, these first results were verified on two further datasets which contain a higher number of movement examples: the stick-throwing 1 and the gesture data. The sick-throwing movements were again simplified by segmenting the movement recordings into building blocks, whereas the gesture data was manually labeled into individual gestures which may contain several building blocks. Additionally, the complexity of the gesture data is higher because examples of the same gesture show a high variance between subjects and the clusters of the classes are more difficult to separate (see section 5.3.3). This makes generalization to new subjects difficult.

In experiment 2, k-NN was compared to HMMs and LSTMs with different hyperparameter settings with respect to classification with small training data sizes. In this experiment, best accuracies were achieved with LSTM and 1-NN classification. On the more heterogeneous gesture dataset an accuracy of 80% was reached with 20 examples per class in the training data, on the simpler stick-throwing data 10 examples per class sufficed for an accuracy of around 90% for both algorithms.

In the third experiment, the generalization to new subjects was evaluated. Here, LSTM classification was better than the two other approaches on both datasets. Especially on the gesture data, LSTM was the only approach which was able to generalize

to new subjects. However, on the simpler stick-throwing data, the accuracy of the 1-NN classification was still above 80%. The complexity of this dataset was reduced by using automatic segmentation into building blocks. Compared to the gesture data, the examples of different subjects of the same movement class are more similar and the movement classes are separated more clearly. That means that simple 1-NN classification achieved good classification results on data which is segmented into its building blocks but cannot generalize well to more complex movements in the case of small training set sizes. This is because using a small number of examples per class in the training data, not all movement variations are available and the determination of the movement class by the closest neighbor in the training data is highly influenced by the selected training examples. In the case of a small number of randomly selected training examples, not every movement variation is covered. This reduced the classification accuracy in the prediction of the movement classes of subjects whose movements where not part of the training data. On the other hand, in LSTM classification a model is learned that can better generalize to movement variations of different subjects in the performed experiments. However, 1-NN had fast calculation times, which makes 1-NN classification a clear alternative to the widely used neural network-based approaches. Additionally, no architecture has to be defined in k-NN and no model needs to be learned. In the presented case of small training set sizes, also the required memory to save the training data for the selection of the nearest neighbor during prediction is low. On both datasets, HMM required more examples to model the demonstrations well enough for a good classification result and had higher computation times.

In summary, the results of the performed experiments indicate that movement trajectories which are segmented into building blocks can be classified with a small number of training examples without using algorithms that are especially designed for few-shot recognition. Possibly due to the simplicity of these movement building blocks, simple 1-NN classification achieves good results in recognizing building blocks of a certain manipulation movement. With increasing movement complexity with respect to the number of movement classes and the variations within each movement class, the usage of a neural network model such as LSTM can improve recognition accuracy. This comes with the cost of increasing calculation times and the necessity to select an appropriate model architecture and to tune hyper-parameters.

6 Chapter 6. Hierarchical Movement Segmentation

In chapter 4, the vMCI algorithm to split observed human manipulation movements into building blocks was introduced and evaluated. The algorithm is able to detect movement sequences with a bell-shaped velocity profile which can be related to primitives of human movements on the level of movement execution, as summarized and discussed in chapter 2. The studies discussed in that chapter indicate, that the generation of human movement is hierarchically organized: On different levels of movement generation, movement building blocks are combined to generate a wide range of behaviors. However, by segmenting human movements using vMCI, building blocks in these movement can be detected but it remains open how these are combined to execute different actions. By detecting the performed actions as well as the building blocks these actions are composed of, a deeper understanding of the human behavior can be achieved. From this, not only LfD applications can benefit, but also other applications such as human-computer interaction or assistance of sick or disabled persons.

In this chapter, methods to automatically split human movement data in a hierarchical manner into meaningful segments are presented. With this, **building blocks** as well as their combinations to more complex movement **actions** should be detected. For this, the vMCI algorithm is extended to a hierarchical approach, which detects building blocks with a bell-shaped velocity and additionally determines the actions consisting of several building blocks by using supervised machine learning methods.

Whereas the vMCI algorithm detects building block movements, such as *approach object*, other approaches in the literature directly detect actions, such as *pour water into cup*, which were defined as movements consisting of several building blocks in section 1.2. Both categories of algorithms detect movements on a single hierarchy level: either building blocks or actions. In contrast to this, the proposed hierarchical approach detects actions and at the same time the building blocks these actions are composed of. An example of an action that should be detected is shown in Figure 3.10, which depicts an asynchronous dual arm movement, in which several single arm building block movements are combined in two different ways to turn a rectangular frame. By detecting both, the building blocks of the movement for each hand, as well has the performed action, not only the single movements' identities are known but also how they can be combined to perform the action of rotating the object. This could, e.g., be used to identify movement parts which a robotic system is already able to perform, teach the system new building blocks, and to provide an execution plan from human example to combine these to execute the rotation task. This approach enables to automatize the analysis of dual arm human movement demonstrations, which is not possible using most of the other segmentation approaches presented in the literature, as summarized in section 6.1.

Two variants of the automated hierarchical segmentation approach are introduced in section 6.2, one based on a small set of labeled training data, and the other one based on clustering approaches. Afterwards an evaluation of the presented approaches on simple, one-handed point-to-point movements executed in a restricted environment is performed as well as an evaluation on several examples of the dual arm rotation movement shown in Figure 3.10, which contains more movement variations. Both datasets were recorded with a marker-based motion tracking system. The work presented in this chapter was published in [Gutzeit, 2022]. Text, figures, tables, and pseudo-code descriptions of the algorithms in the following sections are taken or adapted from that publication.

6.1. Related Work

In the computer vision area, a wide range of algorithms to recognize human actions recorded using different modalities, such as video, RGB or RGB-D data or marker-based approaches have been proposed. A short overview is given in section 5.1. All

these approaches aim to detect movements or actions of a predefined complexity, e.g., whole body movements or postures such as running, walking or jumping [Gong et al., 2014, Kulić et al., 2012] or arm movements, e.g., gestures [Mitra and Acharya, 2007]. Whereas most approaches are based on manual segmentation, e.g., to generate training data, there are several works in which automatic segmentation methods are considered, see the related work of chapter 4 in section 4.1.

An approach which also considers the hierarchical structure of human movements is presented in [Wächter and Asfour, 2015]. They performed a two-level segmentation of 6D pose trajectories of daily activities. First, they semantically segmented the data based on the relation between the human hand and objects with a subsequent sub-division based on the acceleration into so called motion primitives. Also in task planning for robotics, hierarchical approaches based on human movement examples are considered. Agostini et al. presented a hierarchical task planning approach based on schemata, which decomposes robotic movements recorded using kinesthetic teaching [Agostini et al., 2020]. A new movement or task is assumed if the robot enters or leaves a predefined area of the manipulated object or if the gripper is opened or closed. Whereas basic movements, such as picking an object, were learned directly from human demonstration and represented as dynamical movement primitives, the higher-level movements were defined by a human expert. Other approaches, which are based on kinesthetic teaching of basic movement are presented in [Zöllner et al., 2005] and [Caccavale et al., 2019]. However, except from [Wächter and Asfour, 2015], a thorough search of the relevant literature yielded no other work that analysed human movement directly to obtain hierarchically structured movement examples.

6.2. Velocity-based Hierarchical Movement Segmentation

In this section, the extension of the vMCI algorithm to a hierarchical approach which enables the detection of building blocks in human manipulation movements as well as their concatenation to more complex actions is presented. The algorithm, which is named velocity-based Hierarchical Movement Segmentation (vHMS), combines the unsupervised segmentation using the vMCI algorithm with a supervised classification into labeled building blocks and actions.

The basic idea of the vHMS algorithm is to first detect movement building blocks in a data sequence using the vMCI algorithm and subsequently classify these build-

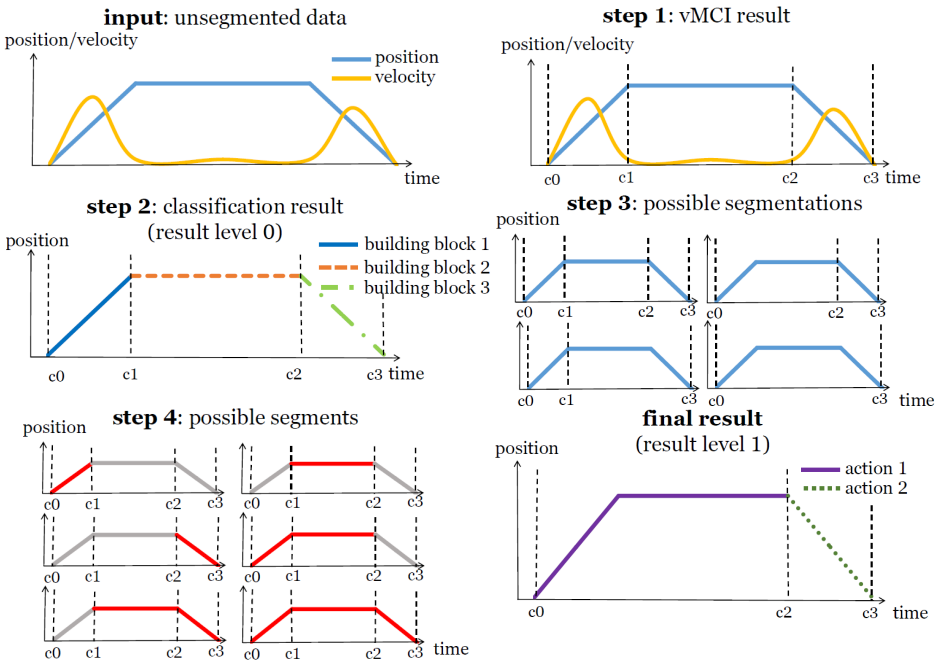


Figure 6.1.: Results for each step of the vHMS algorithm on artificial data. In step 1 and 2, the data is segmented into building blocks with a bell-shaped velocity using vMCI and afterwards classified, resulting in three segments with different classes. In step 3, four possible segmentations are determined, from which the fitness values of six possible segments can be calculated (step 4). The best fitting segmentation is $C_1 = \{c_0, c_2, c_3\}$, consisting of two different actions. *Image extracted from [Gutzeit, 2022], Figure 2.*

ing blocks with an appropriate movement recognition algorithm. Afterwards, the algorithm successively merges segments if their concatenation belongs to a known movement class. These movements, which consist of several building blocks, are referred to as movement actions. These actions can be further merged to more complex actions, until a desired level of complexity is reached. For example, in the movement depicted in Figure 3.10 several building blocks for both hands should be detected using vMCI, e.g., *re-grasp upwards* or *hold*. By combining several of these building blocks of the right and the left hand, the more complex dual arm action *rotate clockwise* should be detected. In this task, different combinations of building blocks can be used to accomplish the same action. In the vHMS algorithm, level 0 refers to a level of low complexity, in which the movement building blocks are detected. With higher levels,

Algorithm 1 Classification-based Hierarchical Segmentation

INPUT movement trajectory $y = (y_1, .., y_T)$ of length T; training data for each level
1. determine building blocks with segmentation points $C_0 = \{c_0, ..., c_{|C_0|}\}, c_i \in \{0, ..., T\}$
2. determine segment labels of $y_{c_i:c_{i+1}}, \forall i \in \{1, .., |C_0| - 1\}$
for all $l \in \{1, ..., max\_number\_of\_levels\}$ **do**
  3. determine all possible segmentations D of C_{l-1}
  4. classify all possible segments A and calculate uncertainty estimate
  5. find best fitting segmentation $C_l \in D$
end for

the complexity of the detected movements increases.

The pseudocode of the vHMS algorithm is given in Algorithm 1. The algorithm is explained using simple artificial data depicted in Figure 6.1. The input for vHMS is a movement data sequence $y = (y_1, ..., y_T)$ of length T, with $y_i = (y_i^{pos}, y_i^{vel}) \in \mathbb{R}^{d+d^v}$ and labeled training data for each level of movement complexity that should be considered in a certain application. In the simple example, the position and velocity are one dimensional. In step 1, the movement trajectory y is segmented using vMCI, resulting in movement sequences with segmentation points $C_0 = \{c_0, ..., c_{|C_0|}\}$, with $c_i \in \{0, ..., T\}$. In Figure 6.1, this results in building block segmentation points $C_0 = \{c_0, c_1, c_2, c_3\}$. The labels of the resulting segments $y_{c_i:c_{i+1}}, \forall i \in \{1, ..., |C_0| - 1\}$, are determined using the training data of level 0. After this step vHMS detected labeled building blocks.

To determine the segments and actions of level 1, the movement segments with segmentation points C_0 are merged into more complex actions in step 3-5. First, all possible segmentations D are determined for the currently detected segmentation points C_0. For the example in Figure 6.1, four different segmentations are possible, i.e., $D_1 = \{C_0, \{c_0, c_2, c_3\}, \{c_0, c_1, c_3\}, \{c_0, c_3\}\}$. For this set of possible segmentations all possible segments A_1 can be extracted. In the simple example $A_1 = \{y_{c_0:c_1}, y_{c_1:c_2}, y_{c_2:c_3}, y_{c_0:c_2}, y_{c_0:c_3}, y_{c_1:c_3}\}$. Using the training data for level 1, labels are assigned to all segments in A_1 and an uncertainty estimate for each segments' classification is calculated. The fitness value of a segmentation is the negative mean value of its segments' classification uncertainty. A segmentation has a high fitness if it is composed of segments, for which the classification has low uncertainty. The

Algorithm 2 Clustering-based Hierarchical Segmentation

INPUT movement trajectories $y^1, ..., y^n$ with $y^j = (y_1^j, .., y_{T^i}^j)$ of length T^j; training data for levels $1, ..., max\_number\_of\_levels$

1. determine building blocks with segmentation points $C_0^j = \{c_0, ..., c_{|C_0|}\}, c_i \in \{0, ..., T^j\}, \forall y^j, j \in \{1, ..., n\}$
2. cluster all segments $y_{c_i:c_{i+1}}, \forall i \in \{1, .., |C_0| - 1\}$

for all $l \in \{1, ..., max\_number\_of\_levels\}$ **do**

3. determine all possible segmentations D^j of $C_{l-1}^j \forall y^j, j \in \{1, ..., n\}$
4. classify or cluster all possible segments A^j and calculate uncertainty estimate
5. find best fitting segmentation $C_l^j \in D^j \forall y^j, j \in \{1, ..., n\}$

end for

best fitting segmentation $C_1 \in D_1$ defines the segmentation points in level 1. In the example, this results in two actions. If desired, step 3-5 can be repeated to generate the segmentation points and labels for actions of a higher level.

6.2.1. Hierarchical segmentation based on clustering

There are applications in which it is not intuitive to select annotations for detected building block segments. For example in some reaching movements, in which the hand of a participant moves towards an object to grasp it and slows down right before grasping, e.g., because the person thinks about the grasping position. This would result in two movement building blocks with no clear class names, because only their concatenation belongs to the class *approach object*. Additionally, using the classification-based vHMS algorithm requires a set of labeled training data for each building block segment and action that should be detected. By using unsupervised approaches in some levels, the manual effort in generating these training data can be reduced. In this section, the clustering variant of vHMS, shortened cl-vHMS, is introduced. For which levels of movement complexity a clustering approach should be used, should be decided depending on the application.

The pseudocode of the clustered vHMS algorithm is given in Algorithm 2. As an input for cl-vHMS, multiple examples of each movement are needed for good clustering results. That means in cl-vHMS n different motion trajectories $y^1, ...y^n, y^j \in \mathbb{R}^{d+d^v}$ are given as input. In Step 1 of Algorithm 2, each trajectory is segmented

individually into its building blocks. Step 2 is replaced by the determination of the clusters compared to step 2 in Algorithm 1, which is done for all building block segments detected in the n movement trajectories. Afterwards, the algorithm is continued in step 3-5 in a similar way as Algorithm 1 by determining the possible segmentations, classifying or clustering of the possible segments and determination of the segmentation which best fits with respect to the uncertainty estimates.

In the experiments, a clustering approach is used in step 2, i.e., steps 3-5 are run on each trajectory individually like in Algorithm 1 using supervised methods. In this way the algorithm still results in labeled actions, but gives only clustered movement building blocks.

6.2.2. Classification and clustering approaches

Different classification or clustering algorithms and uncertainty estimation approaches can be used in the presented hierarchical segmentation algorithms. To minimize the process of parameter tuning, use simple approaches with a small number of parameters should be used. Although in the last years the focus was on deep learning based methods for action recognition, the experiments presented in chapter 5 showed that the simple 1-NN classification is a good alternative for the classification of movement segments detected using the vMCI algorithm, already with a small number of labeled training data. Thus, a simple 1-NN with Euclidean distance is used for the classification of building blocks in step 2 of Algorithm 1. Especially for the few-shot classification of simple movement segments detected by the vMCI algorithm, 1-NN performed better than classification based on HMMs and similar to Long Short-Term Memory networks in the experiments performed in chapter 5. However, by using 1-NN, no hyper-parameters need to be tuned. For the action detection higher levels, the classification algorithm should be selected with respect to the complexity of the actions that should be detected. In the experiments conducted in this chapter, 1-NN is used on both levels. The uncertainty is estimated using the distance of the data sample to its nearest neighbor.

Because of the good results which could be achieved using Euclidean distance based k-NN to classify building blocks, k-means is evaluated for clustering, which clusters a data point to the cluster center which is nearest to it based on the Euclidean distance. The number of clusters k has to be set in advance. To circumvent the selection of k in

k-means, the Gaussian means (g-means) algorithm was proposed [Hamerly and Elkan, 2004]. In the g-means algorithm, the data is incrementally clustered using k-means with $k = 2$ until the clusters are Gaussian. The Anderson-Darling statistic test is used to determine if the resulting clusters are Gaussian or not. If a cluster is not Gaussian, its again clustered into two clusters using k-means. This is done until all clusters are Gaussian or a predefined maximal number of iterations is reached. An adapted version an open source g-means implementation[1] is used to evaluate the clustering of building blocks using g-means in comparison to k-means.

6.3. Experiments and Results

The presented hierarchical segmentation approach as well as its clustering variant were tested on two datasets, the simple one-handed point-to-point movement introduced in section 3.2 and the more complex dual arm object rotation task introduced in section 3.7. The recorded raw marker positions where down-sampled to 30 Hz for movement segmentation. To segment the data using the vMCI algorithm, the 3D positions of the hands were used as input as well as their absolute velocity calculated from these positions. The 3D positions were further preprocessed so that the first order differences are one, as proposed in section 4.3. For classification and clustering, the position of the hand, elbow, and shoulder marker, which were transformed into a coordinate system located at the back of the participants, are used as features. All features were interpolated to the same length and normalized to values between 0 and 1.

In the evaluations, a determined segmentation point within a margin of 0.2 seconds around the true segmentation point was considered as TP. To determine the classification accuracy of the clustering approaches, the unlabeled clusters have to be mapped in an optimal way to the labeled ground truth data. This is a fundamental combinatorial optimization problem called assignment problem, which can be solved using Munkres algorithm [Munkres, 1957]. The Munkres algorithm, also known as Hungarian method or Kuhn-Munkres algorithm, is a method to solve the assignment problem in polynomial time. The Munkres algorithm was implemented in Python based on the matlab implementation available within the BPARHMM toolbox[2].

[1] `https://github.com/flylo/g-means`

[2] `https://emilybfox.su.domains/wp-content/uploads/2021/12/BPARHMMtoolbox.zip`

Table 6.1.: Mean segmentation results of hierarchical segmentation of reference point-to-point movements.

	building block detection		action detection	
	F1-measure (seg.)	ACC (class.)	F1-measure (seg.)	ACC (class.)
vHMS	0.85	**0.79**		
cl-vHMS (g-means)	0.85	0.72	0.77	0.97
cl-vHMS (k-means)	0.85	0.75		

6.3.1. Hierarchical segmentation of point-to-point movements

In the first evaluation of the proposed methods, the simple point-to-point movements recording in the reference setup were used (see section 3.2). In this testbed, the participants had to move a stick through a step pattern with only little space to vary the position, see the right image in Figure 3.4. There are two ways to go through the pattern: starting at the bottom left going to the top right (action 1: *step up*) and going backwards from the upper right to bottom left (action 2: *step down*). The action *step up* consists of two point-to-point movements *left* and *up*, and the action *step down* consists of the two movements *down* and *right*. That means, the movements performed in this simple reference testbed consist of four building blocks (*right*, *left*, *up*, *down*) which can be combined to two different actions (*step up*, *step down*). The movements of all 6 participants, each performing the two actions one after another for roughly 3 minutes, with a short break after each minute, were used for evaluation. In total, 171 examples of each action were available. The whole dataset was manually labeled by the same person by visual analysis of the stick position to obtain a ground truth segmentation as described in section 3.8.

To run the vHMS algorithm a training data set with 4 labeled examples of each action and its building blocks was generated. The examples were randomly selected. The vHMS algorithm was run on the remaining 167 examples. In each example, which consists of the actions *step up* and *step down*, 8 building blocks and 2 actions should be detected, i.e., the correct number of segmentation points is 7 and 1 respectively. The F1-measure of the segmentation and classification accuracy for each algorithm are shown in Table 6.1. The vHMS algorithm detected the building blocks in the test examples with a mean F1-measure of 0.85 with 6 TP and 1.1 FP. The two actions were

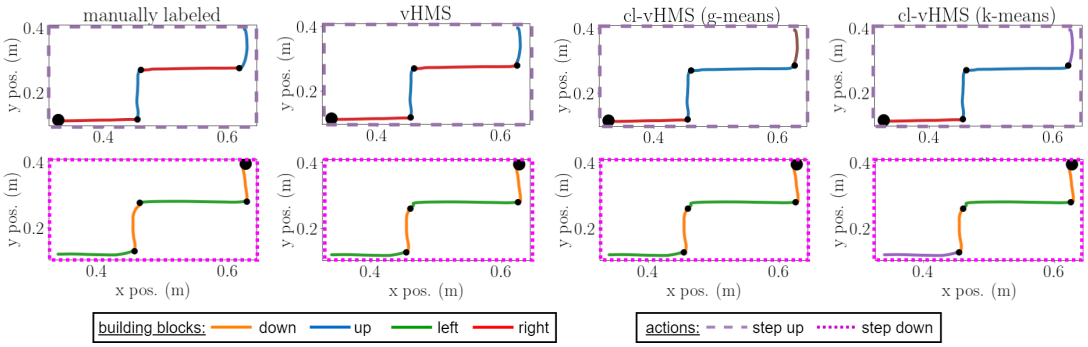


Figure 6.2.: Example segmentation result of the step movement. The hand positions during the movement are shown in colored lines, where the colors indicate the detected building block. The starting position is marked with a big point. The movement in the first row is recognized as action *step up*, the second row as action step down. The result of the manual segmentation is compared to vHMS, cl-vHMS using k-means, and cl-vHMS using g-means. *Image extracted form [Gutzeit, 2022], Figure 4.*

detected with a F1-measure of 0.77, with 0.8 TP and 0.3 FP. The movement labels for each data point were detected with an accuracy of 79% and 97% respectively. That means, that nearly 80% of the data points were assigned to the correct building block, and nearly all these data points were assigned to the correct action.

For the cl-vHMS algorithm, in which 4 examples were clustered at once ($n = 4$ in Algorithm 2), the building blocks were assigned with an accuracy of 72% to the correct class using k-means and 75% with g-means. In the k-means algorithm, k is set to $k = 5$, which is the number of expected building block classes $+1$ to account for movement parts that cannot be assigned to one of the expected movement classes. To determine the accuracy for the clustering approaches, Munkres algorithm was used to map the cluster number to the true labels. An example results of all three variants of the hierarchical segmentation compared to a manual segmentation is shown in Figure 6.2. In this evaluation the approaches obtain good results based on a small number of movement examples also if no labeled training data is used for the detection of building blocks.

6.3.2. Hierarchical segmentation of of two-handed object rotation movements

The second evaluation was done on a dual arm rotation task introduced in section 3.7. The participants were instructed to grasp a rectangular frame with both hands, lift it to a horizontal position, turn it to vertical position and back to horizontal position, and finally placing it back on the table. The recorded movements consist of four different dual arm actions *lift*, *turn clockwise*, *turn counterclockwise*, and *place*, with 7 building block segments for each hand: *lift*, *re-grasp down*, *re-grasp up*, *hold*, *turn clockwise*, *turn counterclockwise*, and *place*. All movements recorded, in total 33 examples for each action, were used for evaluation. This dataset is well suited for the evaluation of the hierarchical segmentation because it contains dual arm movements with synchronous as well as asynchronous movements of the hand. Additionally, the turning actions were executed with different compositions of building blocks, see for example the two variants of *turn clockwise* visualized in Figure 3.10.

To run the vHMS algorithms on the dual arm data, training data consisting of 8 randomly selected examples of each movement was generated. Each training example contained the labeled building blocks for the right and left hand respectively, as well as the data belonging to the whole action with the dedicated label. To segment the dual arm data hierarchically, step 1 and 2 of Algorithm 1 and 2 were run for each hand separately, so that the resulting building blocks could clearly be assigned to the correct hand. In the vMCI algorithm, it is also possible to evaluate the data of both hands at once. However, this would result in building block segments which end at positions where both hands finish a sequence with a bell-shaped velocity at the same time, which is not always the case for the asynchronous dual arm movement analyzed in this experiment. In the cl-vHMS algorithm using k-means, the parameter k was set to 8, which is again the number of expected building blocks $+1$. For both clustering variants 8 examples were clustered at once.

For this data, the true number of segmentation points can differ between movement examples because of the different possibilities to fulfill the task. On average, the movement examples contain 16.4 building blocks and 5.6 actions. The results of the automatic segmentation and classification can be seen in Table 6.2. The segment borders of the building blocks could be detected with a mean F1-measure of 0.74, whereas the F1-measure for the result on one example was determined by summing up the mean of

Table 6.2.: Mean segmentation results of hierarchical segmentation of dual arm rotation movements.

	building block detection		action detection	
	F1-measure (seg.)	ACC (class.)	F1-measure (seg.)	ACC (class.)
vHMS	0.74	0.62		
cl-vHMS (g-means)	0.74	**0.70**	0.72	0.80
cl-vHMS (k-means)	0.74	0.65		

the results of the individual hands. The mean number of TP was 10.4 and the number of FP was 3.6. Using vHMS, the building block segments were assigned to the correct class with an accuracy of 62%. Using cl-vHMS, the point-wise accuracy was 70% for the g-means approach and 65% for the k-means approach. An example result can be seen in Figure 6.3. For this dataset, the clustering approaches, which do not require a set of labeled examples of the building block movements for training, performed better than vHMS. This is, because the dataset contains unlabeled movement parts which cannot be assigned to one of the expected movement classes. Using vHMS with 1-NN, these parts were mapped to a known class which automatically results in a miss-classification. Using unsupervised methods, these movements can be grouped to a new cluster, which is mapped using Munkres algorithm to the ground truth class with no label. With these methods, nearly 3/4 of the data could be assigned to the correct building block movement using g-means based cl-vHMS, despite the clear drop of segmentation accuracy compared to the results in section 6.3.1. The segment borders of the dual arm actions were detected with a F1-measure of 0.72, with 2.6 TP and 0.3 FP, and a point-wise accuracy of 80%.

6.4. Discussion

The vHMS approach for hierarchical segmentation of human manipulation movements was presented in this chapter. The algorithm automatically detects movement building blocks with a bell-shaped velocity profile which are assumed to be characteristic for human manipulation movements as well as movement actions which are composed of several building blocks. The experiments, performed on simple one-handed point-to-point movements and a more complex object rotation task which

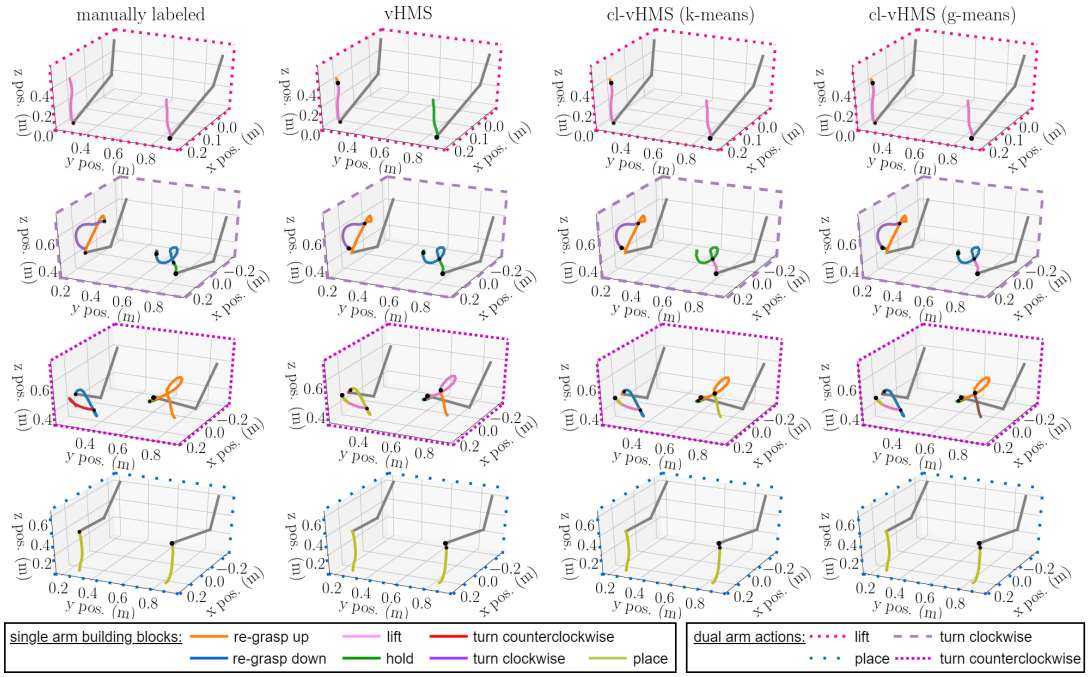


Figure 6.3.: Hiearachical segmentations of a rotation movement in comparison to manual segmentation. Example segmentation result of a rotation movement. Shown are the actions *turn clockwise* (first row) and *turn counterclockwise* (second row). The lines between hand, elbow, and shoulder marker are shown in grey to visualize the arm position. The hand positions are shown in colored lines, where the colors indicate the detected building block of each hand. The results of the manual segmentation are compared to vHMS, cl-vHMS using k-means, and cl-vHMS using g-means. *Parts of these plots were originally published in [Gutzeit, 2022], Figure 5.*

consists of synchronous as well as asynchronous dual arm movements, show that the algorithm reliably detects building blocks and actions in movement examples of several subjects using a small number of labeled training examples.

Furthermore, a variant of the hierarchical segmentation was presented, cl-vHMS, in which the supervised classification of building blocks movements is replaced by unsupervised approaches which do not require a manual generation of training data. For the dual arm movements, which contain small extra movements that cannot be clearly assigned to one of the defined movement classes, the cluster-based method using g-means outperformed the supervised vHMS approach by approximately 10%.

The proposed hierarchical algorithm is designed in a way that the classification and

clustering methods can easily be replaced. Using for example neural network based approaches may improve action detection accuracy, also if more complex actions are analyzed. For the building block detection, simple approaches such as the proposed 1-NN classification achieve good results, because the building blocks are by definition simple movement identities.

Part III.

Applications in Robotics

7 Chapter 7. Application in Robotic Learning from Demonstration

In this chapter, a framework to transfer human movement examples to robotic systems using imitation learning is introduced. By using the segmentation and recognition approaches described in the previous chapters, movement building blocks can be detected in human demonstrations and directly be used to generate robotic behavior using this framework. Using this, a robotic system can learn solutions of tasks that were not foreseen during design without robotic expert knowledge. This is required if the system should be deployed in unknown or unpredictable, dynamic environments, e.g., in space, search and rescue, and underwater scenarios.

The imitation learning framework is called the BesMan Learning Platform. It is a modular framework that incorporates all steps needed to generate robotic behavior from demonstration using imitation and reinforcement learning, including the automated approaches to segment and annotate human movement demonstrations. Complete descriptions of robot skill learning frameworks are hardly present in the literature. A thorough search of the relevant literature yielded only one related article published by [Peters et al., 2012] that gives a complete overview of a learning architecture and is comparable to learning platform presented in this chapter. Their work includes imitation learning and reinforcement learning methods to learn so-called motor primitives as well as generalization methods for these motor primitives and even describes methods to learn operational space control. However, in their work as well as in the majority of similar works the relevant behaviors are directly presented by

kinesthetic teaching so that there are no kinematic and dynamic differences between the demonstrator and the target system. In addition, only the relevant behavior is presented or it is not discussed how the relevant part that should be transferred is extracted. In contrast to that, the goal of the imitation learning framework presented in this chapter is that a human can demonstrate the behavior as naturally as possible. By integrating unsupervised segmentation methods, the relevant movements in the demonstrated behavior can be detected automatically. In the BesMan learning platform, the vMCI algorithm is used to detect the movement parts in the human demonstration which should be transferred to the robotic system. As described in chapter 4, the vMCI algorithm detects building blocks in human manipulation movements. By transferring these building blocks to a robotic system, central entities for manipulation tasks are provided to the system, which can be combined to solve also similar tasks which were not directly demonstrated by the human.

In section 7.1, the individual modules of the behavior learning framework are explained. Afterwards, the framework is evaluated with respect to automation level and time requirements needed to teach a robotic arm to throw a ball. Additionally, two applications on other learning tasks are presented. In a further experiment, the generalization ability of the framework is shown by transferring a large set of stick-throwing demonstration to four different robotic systems.

The texts, figures, and tables in this chapter are taken or adapted from [Gutzeit et al., 2018a] and [Gutzeit et al., 2019]. The general concept of the learning was first published in [Metzen et al., 2013].

7.1. BesMan Learning Platform

The framework for robotic movement generation using human demonstrations was developed within the project BesMan[1]. The contribution of this thesis to this framework is the acquisition of the human movement demonstrations and the integration of processes to automatically segment and annotate the recorded movements to detect the relevant parts which should be transferred to the robotic system. In this section, the whole learning platform is presented, including a short overview of the learning algorithms needed to generate robotic behavior from the acquired human movement demonstrations.

[1] For more details, see `http://robotik.dfki-bremen.de/en/research/projects/besman.html`

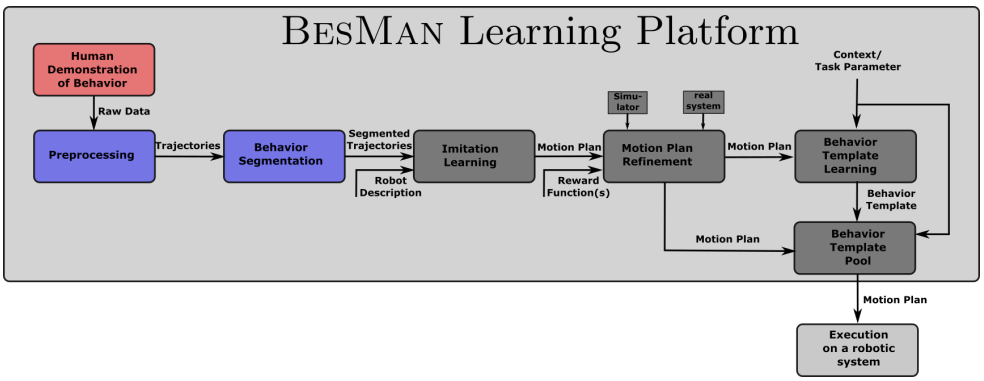


Figure 7.1.: Dataflow diagram of the BesMan Learning Platform. A behavior is demonstrated and segmented into smaller behavioral building blocks. Motion plans corresponding to these building blocks are imitated and refined using reinforcement learning and/or transfer learning. Acquired motion plans are specific for a task but can be generalized to more generic behavior templates. Once a new specific task is encountered, the behavior template is instantiated and yields a task-specific motion plan. The modules which are presented in detail in this thesis are highlighted in red and blue. *Image adapted from [Gutzeit et al., 2018a], Figure 1.*

An overview of the BesMan learning platform can be seen in Figure 7.1. It consists of several modules needed to generate so-called motion plans from human movement demonstrations which can be executed on a robotic system. Motion plans represent solutions to generate specific behaviors. Furthermore, more generic behavior templates can be generated. Behavior templates represent generic movements to generate a flexible behavior able to, e.g., reach different points in space and to be executed on different systems with different morphology.

To generate robotic behavior from detected movement segments, the human motion needs to be mapped to the robot morphology with a subsequent motion refinement, which ensures that the desired goal is reached. The three main modules of the learning platform consist of the acquisition and selection of appropriate human movement examples, the imitation of these and the refinement of the imitated movement. They are described in more detail in the following paragraphs.

7.1.1. Acquisition of human movement examples

Motion tracking data, such as the datasets described in section 3, can directly be used as input to generate robotic movements using the learning platform. In the

framework, the vMCI algorithm for automated movement segmentation as well as 1-NN for segment classification are directly integrated. As evaluated in detail in chapter 4, vMCI reliably identifies building blocks in human manipulation movements without manual user input. To generate labels for the detected segments, the 1-NN classification is a suitable method which only requires a small number of labeled training examples, as evaluated in detail in chapter 5. Using this, the segments that should be learned and transferred to the robot can be automatically selected from the recorded movement data.

To run the automatic segmentation and recognition, it is important that the recorded maker positions can be transferred into a coordinate system which is local to the demonstrator. Using the Qualisys system for data recording, as described in detail in section 3.1.1, this is realized by placing three markers at the back of the subject, from which the local coordinate system can be determined. After the data is down-sampled, it can directly be used as input for the vMCI movement segmentation with a successive movement classification. The same applies for movement demonstration recorded using the Xsens motion suit, for which reference positions at the back of the subjects can be used to transfer the data into a local coordinate system. For each recording modality, at least the position and the velocity of the hand of the demonstrator needs to be recorded, because the vMCI algorithm uses this information to detect movement building blocks, as described in detail in chapter 4.

7.1.2. Imitation learning

Each relevant segment that was detected using vMCI segmentation and 1-NN classification is represented as a motion plan using imitation learning. Motion plans describe trajectories that can be executed by the robot and mimic the trajectories presented by the human demonstrator during a single behavioral building block.

Due to the different morphology of humans and systems, human behaviors cannot be directly transferred. Instead, the correspondence problem has to be solved, i.e., the problem on how human body positions can be mapped to parts of the robotic system to generate executable motions [Nehaniv and Dautenhahn, 2002]. To solve the correspondence problem, a record mapping is needed which maps marker trajectories of human demonstrations to a sequence of actions or system states, as well as an embodiment mapping, which maps the recorded sequence to a trajectory that can

be executed on the target system [Argall et al., 2009]. As human joint angles cannot be directly measured using motion tracking the record mapping extracts the human hand pose trajectories from the recorded marker positions and represents them in a coordinate frame local to the human subject. The embodiment mapping makes the trajectory executable on the target system, despite possible difference in the workspace, kinematic structures and dynamic capabilities between human teacher and system. In the learning platform the embodiment mapping is defined automatically by using black box optimization in which reachability of the desired poses are optimized, while minimizing the risk of getting too close to singularities, avoiding self-collisions, and maximizing exploitation of the robot's workspace. For the optimization, any black box optimizer can be used. A detailed description of the record and embodiment mapping can be found in [Gutzeit et al., 2018a] and [Gutzeit et al., 2019].

After the recorded trajectory is mapped to the robot's workspace, a suitable representation that can be used for further adaptation and refinement is needed. A popular class of policy representations that has been used to learn movement primitives for robotic manipulation are DMPs [Ijspeert et al., 2013, Pastor et al., 2009]. With a DMP the robotic movement is represented as a nonlinear dynamical system which converges to a given target. By adjusting the weights of the forcing term in the DMP formulation to the measured movement demonstration, the shape of the trajectory can be regulated to achieve the desired behavior.

There are numerous advantages of DMPs in comparison to other policy representations for imitation learning, among them: 1. They are stable trajectory representations. Slight errors in execution of the trajectory will not result in error accumulation like in general function approximators. 2. To reproduce a demonstrated movement, a one-shot learning algorithm can be used that determines the weights of the forcing term. Hence, imitation learning with DMPs is much simpler than it is for more general function approximators. 3. Movements can be easily adapted (even during execution): the goal of the movement can be changed and obstacles can be avoided.

7.1.3. Motion refinement

Although the record and embodiment mapping described in the previous section ensure that the movement is executable on the target system, the generated motion plans might not produce the same result as the human demonstration. For example, a

ball thrown using an imitated throwing building bock that is executed on the target system might not hit the same position like in the human demonstration. This results from considerably different kinematic and dynamic properties between human and robot. To account for this, reinforcement learning methods are used to refine the imitated motion plan. There are several policy search methods integrated in the learning platform, such as Covariance Matrix Adaption Evolution Strategy (CMA-ES; [Hansen and Ostermeier, 2001]), Relative Entropy Policy Search (REPS; [Peters et al., 2010]), and Path Integral Policy Improvement (PI^2; [Theodorou et al., 2010]). All integrated algorithms require an interaction with the real or simulated target system and the specification of a reward function which tells the learning algorithm how well a motion plan solves the task.

Alternatively or in conjunction with reinforcement learning, transfer learning can be used to adapt motion plans. Using this, differences in the execution of motion plans in simulation and on the real system are considered during motion plan refinement. To account for this so-called simulation-to-reality-gap, the transferability approach [Koos et al., 2013] is integrated into the learning platform. The approach minimizes the number of tests on the real system by finding motion plans which are maximizing the reward in simulation as well as the transferability of these motion plans to the real world.

The resulting motion plans are solution for the specific setting the human demonstrated the movement in. The learning platform also includes contextual policy search methods [Deisenroth et al., 2013] in the behavior template learning module, to generate more general representations of these motion plans, so-called behavior templates, which can be executed in different but similar settings. More details about the methods for motion refinement and behavior template learning can be found in [Gutzeit and Kirchner, 2016] and the references contained therein.

7.2. Experiments and Results

The learning platform was tested and evaluated on different tasks and robotic systems, which are presented in this section. In 7.2.1, the learning platform is evaluated with respect to time requirements and level of automation on ball-throwing movements which were transferred to the robotic arm COMPI [Bargsten and de Gea Fernández, 2015]. The transfer of grasping movements to the Kuka iiwa lightweight robot and

Figure 7.2.: Robotic arm COMPI with a spoon mounted as end effector. The ball that should be thrown was placed into the spoon. *Image adapted from [Gutzeit et al., 2018a], Figure 3.*

lever-pulling movements to the robotic system Mantis [Bartsch et al., 2016] are presented in section 7.2.2. Both of these experiments were originally presented in [Gutzeit et al., 2018a] and are summarized in this section. Afterwards, the generalization ability of the learning platform to different systems is presented by transferring stick-throwing movements to different robotic system, an evaluation which was originally published in [Gutzeit et al., 2019].

7.2.1. Transfer of ball-throwing movements to COMPI

In this section, the learning platform is evaluated as a whole in a ball-throwing scenario. Human demonstration of ball-throwing movements are transferred to the robotic arm COMPI, which is displayed in Figure 7.2. A scoop that can hold a ball is mounted as COMPI's end effector.

The ball-throwing movements described in section 3.5 are used as demonstrations for this evaluation. The datasets consist of throwing demonstrations of 10 differed subjects, each performing 24 throws. Due to a high number of occlusions during movement recording, the marker positions were not continuously tracked. An automatic labeling approach, which assigns labels to trajectories of the passive markers of the Qualisys system based on the relative positions of the markers to each other, is tested on this dataset. In the evaluation this approach is evaluated with respect to time requirements to manual marker labeling.

The recorded data was automatically segmented using vMCI with successive an-

notation using 1-NN into the classes *strike out*, *throw*, *swing out*, and *idle*. To train the 1-NN classifier, a training set of 40 throwing demonstration from subjects 1-5 (8 throwing examples from each subject) was generated. The throwing movements of subjects 1-5 that were not used for training and all the throwing movements of the subjects 6-10 were used to evaluate the classifier. On the test set consisting of the remaining demonstrations of subjects 1-5, an accuracy of 93.2% could be achieved. The important movement class `throw`, which should be transferred to the robotic system, was detected with an accuracy of 98.6% on this test set. The second test set consisted of demonstrations of subjects 6-10, i.e., no throwing movements of these subjects were used to train the classifier. On this data, the accuracy was 89.1% and the class `throw` was detected with 93.9% accuracy. As a conclusion, different movement classes can be successfully identified using vMCI segmentation and 1-NN classification.

To transfer an identified throwing building block to the system, the trajectory is translated and rotated so that it fits into the workspace of COMPI. The end effector trajectories are transformed into joint trajectories via inverse kinematics. In addition, the joint trajectories are scaled to respect the joint velocity limits of the target system. In a last step, the motion plan in generated by representing the throwing movement building block as a joint space DMP via imitation learning. Moreover, a minimum execution time of 0.95 seconds is set to reduce the velocity and accelerations, which are penalized during the subsequently performed optimization of the motion plan.

Due to kinematic and dynamic differences between the human demonstrators and the robotic system COMPI, an imitated throwing movement has not the same effect on the ball as the human movement. For this reason, the motion plans are adapted using the transferability approach, as described in detail in [Gutzeit et al., 2018a]. Several parameters including the initial position, the goal position, DMP weights, and execution time are optimized to generated throwing-motion plans which result in a touchdown position of maximal 10 cm around the target position. The target distance of the touchdown position of the ball is optimized in simulation, with intermediate execution of promising throws on the real system.

Results An overview of the time required for each step needed to transfer the throwing movements can be found in Table 7.1. The required time for successful automatic marker labeling is much faster than manual labeling even though the automatic labeling is slow because of the high number of gaps in the recorded marker trajectories.

Table 7.1.: Required time (per experiment, 8 throws) for each stage of the learning platform. Table adapted from [Gutzeit et al., 2018a], Table 1.

Step	Time (min)	Auto-mated	Required knowledge
Attaching markers	0:55	✗	
Motion capture	1:08	✗	
Automatic marker labeling	4:58	✓	Neighboring markers, initial pose
Manual marker labeling	9:19	✗	
Behavior segmentation	0:44	✓	
Manual movement labeling[2]	50	✗	
Movement classification	0:02	✓	
Imitation learning	4:20	✓	Robot description
Policy search	10	✓	Reward function, simulation
Transferability approach	75	(✓)	Reward function, simulation

[2] training data sets contains 40 throws

If the markers were always visible the automatic labeling would have taken only about some seconds. The manual movement labeling, which is needed to generate the training dataset consisting of 40 example throwing demonstration, is very time intensive but has to be done only once. In the transfer of further throwing movements, the same training set can be used. Furthermore, the experiments in section 5.4 showed that good classification accuracies can be achieved if this training dataset is reduced to less than 20 throwing examples, which would also reduce the time required to generate the training dataset. The longest part in the whole process is the refinement for the target platform (imitation, policy search, transfer), which is a difficult problem that involves interaction with the real world.

Although, the process of acquiring new behaviors is automated, still some human intervention is required either through knowledge that has to be given to the system or by interacting physically with the system. An overview can be found in Table 7.1. Of course it is necessary that a human demonstrates the movement. The labeling of the markers can be completely automated with more cameras. However, the maximum possible level of automation was not achieved in the presented experiments.

Movement classification requires a dataset that is labeled manually but this effort can be minimized by using a classifier which recognizes the movements at a high accuracy with small training data sets. When the learning platform is set up for a new target system and type of manipulation behavior, the components that are combined for embodiment mapping has to be selected. However, the imitation learning is completely automated. The motion plan refinement with the transferability approach requires human assistance because the robot has to try throwing movements in the real world and is not able to get the ball back on its own. The process itself is automated so that no knowledge about the system or the task is required from the human at this step. In addition, a reward function has to be defined that describes how a solution of a task should look like and a simulation has to be prepared to minimize the interaction with the real world. These are manual processes at the movement.

7.2.2. Application of the learning platform on different movements

Next to the transfer of ball-throwing movements to the COMPI, the learning platform was used to transfer a grasping movement and a lever-pulling movement to a Kuka iiwa lightweight robot and the robotic system Mantis [Bartsch et al., 2016] respectively. The systems can be seen in Figure 7.3 and videos of both applications can be found online.[3]

In the transfer of a grasping movement, the Kuka robotic arm should grasp a box from a shelf. The grasping movement was imitated from the human pick-and-place demonstrations described in section 3.4, in which a box is grasped from a shelf, placed on a table, and put back afterwards. After successful detection of a grasping movement building block using vMCI segmentation and 1-NN classification, it was imitated using a Cartesian space DMP and adapted to be executed on a Kuka robot equipped with a 3-finger gripper from Robotiq. Only one demonstration was required to learn the grasping movement. Cartesian DMPs were used for easy integration with the used whole-body control and perception. In this scenario the refinement was done using the CMA-ES algorithm in simulation. After 50-100 iterations, the movement could be successfully transferred to the robotic system.

[3]Grasp box: `https://robotik.dfki-bremen.de/de/mediathek/videoarchiv/besman-zweite-demo.html`; pull lever: `https://robotik.dfki-bremen.de/de/mediathek/videoarchiv/besman-dritte-demo.html`

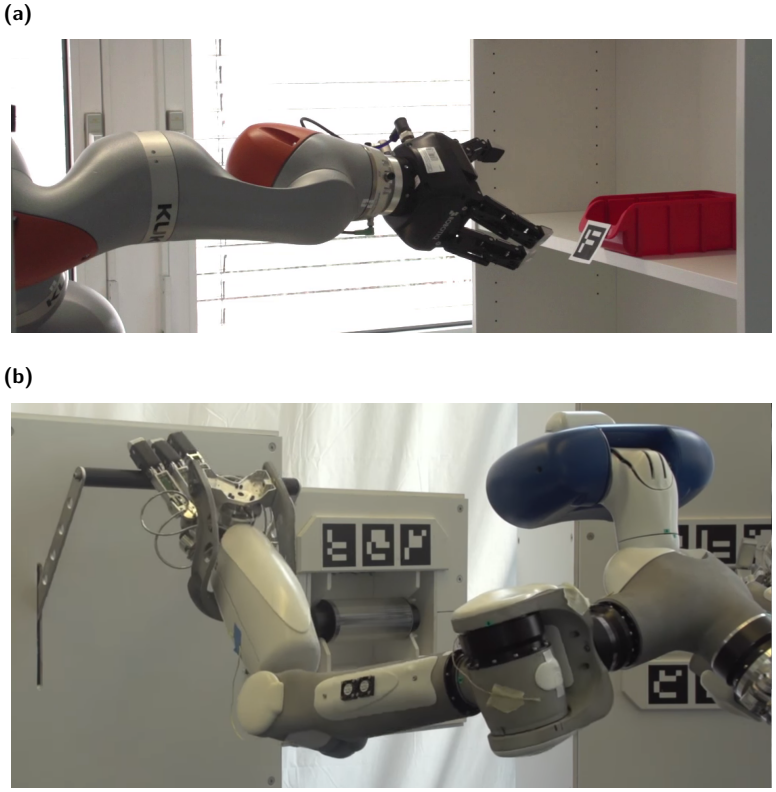


Figure 7.3.: Excerpts from two movement executions which were taught to the systems using the BesMan learning platform. (a) Kuka iiwa lightweight robot grasping a box from a shelf. (b) Mantis robot executing a lever-pulling movement.

In another scenario, the learning platform was used to teach the robotic system Mantis to pull a lever. For this, the movement demonstrations presented in section 3.3 were used. Again, the recorded movements could be successfully segmented. Due to the fixed lever position in this experiment, the movement execution of the human was strongly predetermined leading to good classification results with only one training example per class, see section 5.4. Like in the pick-and-place scenario, reinforcement learning techniques implemented in the learning platform were used to adapt the demonstrated movement to the robotic system. REPS and CMA-ES gave good results. After several hundred episodes in simulation, a successful movement could be generated. Learning could be done in parallel from multiple demonstrations with each RL learning process being initialized with a single demonstration.

As in the ball-throwing scenario, the transfer of the demonstrated movements was partially automated in the transfer of grasping and lever-pulling movements. To recognize the important movement segment, only a few manually labeled training examples are needed. To imitate and adapt the demonstrations to the system, the embodiment mapping and a reward function have to be selected with regards to the robotic system and the task goal. Besides this, the transfer of the movement to the system is completely automated.

7.2.3. Transfer of stick-throwing movements to different systems

In the experiments described in this section, stick-throwing movements were transferred to four different robotic systems in simulation and one of these systems in real world without refinement of the imitated trajectories. For this evaluation, motion plans for these systems are generated using human movement demonstration of the two stick-throwing datasets described in section 3.5, consisting of 697 and 34 movement examples respectively. The imitation of the movements was performed with a configurable version of the learning platform, which allows to easily adapt the learning for different robotic system by optimizing a generalized goal function in the embodiment mapping, as described in more detail in [Gutzeit et al., 2019].

Transfer in simulation The generality of the movement segmentation and classification as well as the transfer to different robotic systems, namely Universal Robots' UR5 and UR10, KUKA iiwa lightweight robot, and COMPI, was evaluated on the stick-throwing dataset 1, containing 697 movement demonstrations. To evaluate the movement classification, a stratified cross-validation repeated 100 times was performed with a fixed number of examples per class in the training data. Furthermore, the number of successfully transferred movements as well as the difference between the position of the hand in the demonstrations and the end effector position of the systems are analyzed.

The automatic segmentation of 697 demonstrated throws resulted in 2913 detected segments with a bell-shaped velocity profile. Although some throwing demonstrations showed just a small decrease of the velocity of the hand between throw and swing out phase of the movement, most of the throwing segments were successfully segmented. As an example result, a demonstration of one subject containing 41 throws is visualized

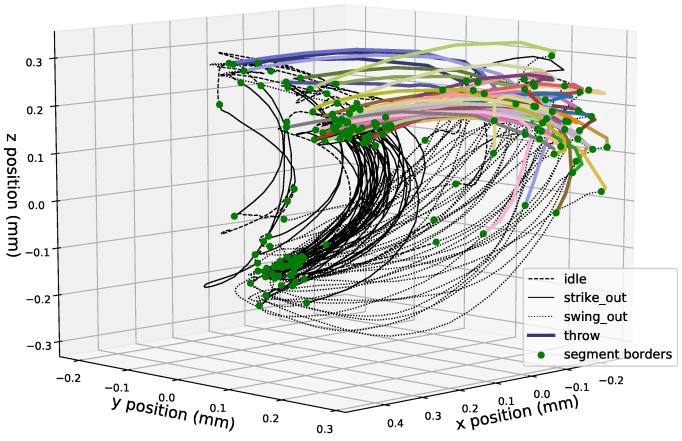


Figure 7.4.: Throwing trajectories of subject 5. The position of the hand is shown for one set consisting of 41 stick-throwing movements. Green dots mark the result of the vMCI segmentation. The resulting segment trajectories were labeled with 1-NN classification. Different line styles mark the different classes. Throwing segments are visualized as straight lines with a different color for each throw. *Image extracted from [Gutzeit et al., 2019], Figure 3.*

in 7.4.

2233 of the detected movement segments could clearly be assigned to one of the movement classes *strike out*, *throw*, *swing out*, and *idle* and were used to evaluate the annotation. The number of training examples per class was varied between 1 and 20. With 4 examples per class a mean classification accuracy of 90% could be achieved. 95% could be achieved with 9 examples per class. Thus, a training data set with 9 examples per class was created, which contains the first three throwing demonstration of three different subjects. Using this training data, the segments of all recorded demonstration were classified. The movement class *throw* could be detected with an accuracy of 99%, with 623 correctly detected throwing movements, 13 false negatives and 2 false positives. In 7.4, different line styles indicate the resulting labels in the movement demonstrations of subject 5. The segments of this subject were classified with an accuracy of 98%, with 24 segments that could not be clearly assigned to one of

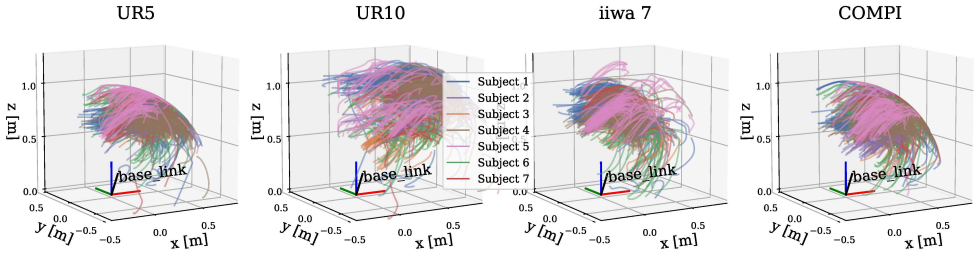
the four movement classes and was thus removed from evaluation. This result shows that the approach to detect the relevant parts in the demonstrations also generalizes to large datasets. A small number of labeled training data was sufficient to annotate the automatically derived segments.

The throwing trajectories mapped into the workspace of the robotic system UR5, UR10, KUKA LBR iiwa, and COMPI, are shown in Figure 7.5 (a). 682 trajectories were transferred to the workspaces of all target systems. Most of the trajectories easily fit in the workspace of UR10 (arm radius: 1300mm) and KUKA LBR iiwa 7 (arm radius: 1266mm), while many trajectories are distorted or are close to the borders of the workspace of UR5 (arm radius: 850mm) and COMPI (arm radius: 940mm). Throwing movements often tend to be close to the borders of the human's workspace. Hence, the selected skill is quite challenging for smaller robots. Figure 7.5 (b) shows the demonstration of subject 5 visualized in 7.4 transferred to the robotic arms. The different colors match the colors of the throwing segments in 7.4. Figure 7.5 (c) shows the ground contact points of sticks for the presented throwing trajectories from simulation. It can be seen that on average the UR10 has the widest distribution as it has the largest workspace. In the next section, it is quantified how well the throwing trajectories can be transferred to the real UR5 robot, as it is one of the more challenging robotic systems due to the more restricted workspace.

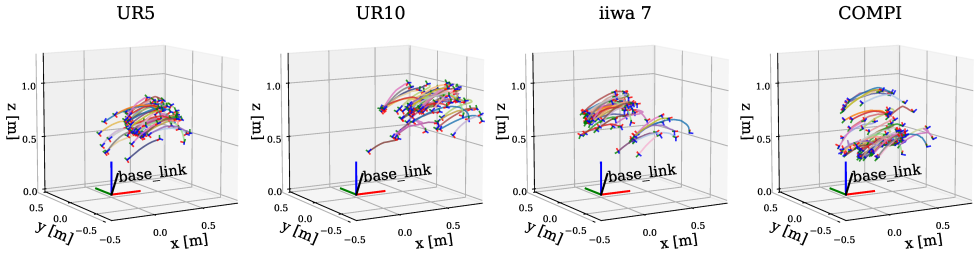
Transfer to a real system The quality of the transferred movements is evaluated by comparing the trajectory and goal position of the stick thrown by humans to the one thrown by a real UR5 robot. For this, the stick-throwing set 2 was used, in which also the position of the stick during the movement was tracked. Additionally, the movement of the robotic arm UR5 is captured by placing a marker at the end effector, see Figure 7.6.

The 34 demonstrations were transferred to the real UR5 robotic arm. The number of successful throws was analysed, the stick position during the throw, and its goal position. The transferability of the demonstrated throws on the UR5 robot is evaluated with respect to the following aspects: Does the robot inadvertently collide with anything including the stick? Does the stick fall out of the stick holder while the robot approaches the starting pose of the trajectory? Does the stick leave the holder during the throwing movement? If any of these aspects are evaluated negatively, the trajectory is considered not transferable. To evaluate the quality of the transferred

(a) All recognized throwing movements transferred to the workspace of the four robots. Trajectories from the same subject are shown in the same color



(b) All throwing movements of subject 5. Colors indicate the indices of the throws and correspond to the colors in. The frames of the robots' base links are shown.



(c) Throwing results in simulation. The distribution of ground contact points of the sticks is displayed. Colors of the points indicate the index of the transferred throwing trajectory

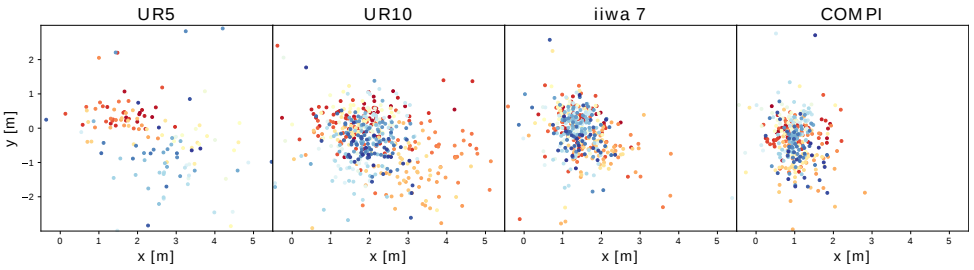


Figure 7.5.: End effector trajectories of throwing movements in robots' workspaces and corresponding ground contact points of the sticks. *Image extracted from [Gutzeit et al., 2019], Figure 4.*

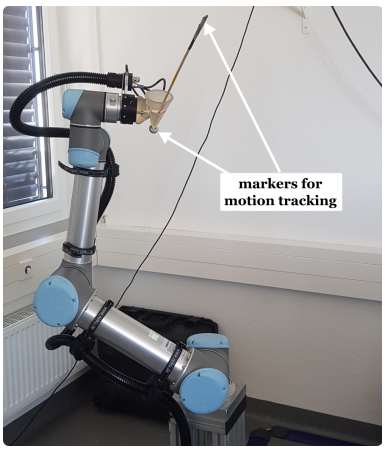


Figure 7.6.: Movement recording setup. To record the stick position, a marker is attached to the tip of the stick. The movement of the robotic arm UR5 is tracked with a marker attached to the end effector. *Image extracted from [Gutzeit et al., 2019], Figure 2 (b).*

throws, the stick trajectories of the demonstrated throws and the recordings of the throws transferred to the UR5 were compared. Since the motion capture system sometimes returned noisy stick position measurements, the trajectories had to be interpolated. A quadratic model was used for interpolation. The same model was used to extrapolate both the demonstrated and the reproduced stick trajectory until the stick hit the ground. Before this, the demonstrated trajectory was aligned to the start position of the transferred one. Thus, the distance between ground contact points as well as the similarity of the the stick trajectories can be determined. To compare the trajectories, the average DTW [Sakoe and Chiba, 1978] distance was used, i.e., the DTW distance divided by the maximum number of steps of the two time series.

In this experiment, the throwing movements were detected with an accuracy of 97%, using the same training data as in the first experiment. 33 throws were correctly detected and one was wrongly assigned to another class by the 1-NN classification. 27 out of these segments could be transferred to the real UR5. The executed movements showed useful throws. However, the goal positions of the demonstrated throws are not reached by the system. The biggest difference between demonstrated goal position and goal position of the imitated movement was nearly $1.2m$. The mean goal distance was $0.72m$ (standard deviation: $0.31m$). The mean average DTW distance between the demonstrated and executed trajectories were $0.15m$ (standard deviation: $0.1m$).

The results show that it is possible to automatically imitate demonstrated throws and that most of these throws are executable on the real system but the goal position of the demonstrated movement could not be reached by solely using imitation learning without motion refinement.

7.3. Discussion

The results show that it is possible to learn new skills for robots without specifying the solution directly. The learning platform leverages intuitive knowledge from humans that do not know anything about the target system to automatically transfer skills to robots. By integrating automatic movement segmentation and recognition approaches as introduced in chapter 4 and 5, building blocks can be identified in human movement demonstration and used as a basis to learn new robotic behavior. The main impediments that can be overcome by the learning platform in this setting are the kinematic and dynamic differences of the demonstrator and the target system.

In the transfer of ball-throwing movement, is was shown that new robotic behavior can be learned with only minimal user interference in a reasonable time period. Additionally, a reaching and lever-pulling movement building block could be used as movement example for imitation and resulted in successful movement executions on a robotic system. Especially the successful transfer of the reaching movement is interesting because this movement building blocks is part of the pick-and-place movements on which the vMCI algorithm showed the lowest segmentation performance in the evaluations of section 4.4 resulting from inaccurate segment borders. Nonetheless, a successful reaching movement could be learned using the learning platform.

In a third experiment, stick-throwing trajectories were automatically extracted from human demonstrations and transferred to four robotic target systems. The experiment showed that the embodiment mapping, which is needed to map human movement trajectories into the robot workspace, can be automatized for a dataset of 697 throws. Throwing is a challenging skill for these robots because it has high acceleration and velocities and is close to the border of the workspace of humans. Nonetheless, most of the demonstrated throws could be transferred to the systems using the proposed framework. Furthermore, the difference of stick trajectories and ground contact points between demonstrated throws and reproductions of those on a real UR5 were evaluated. Although the demonstrated throwing movements could

be successfully executed on different systems, there is still a significant gap between the outcome of demonstrated throws and their reproductions. This could be solved by using reinforcement or transfer learning to refine the movements, as done in the transfer of ball-throwing movement in the first experiment.

The presented approach still has some limitations: some prior knowledge has to be defined in form of reward functions, simulations, and preparation of markers for motion capture. For complex problems with complex reward functions, learning the reward function would be better than defining it manually. Promising fields of research are active reward learning [Daniel et al., 2015] and inverse reinforcement learning [Ng and Russell, 2000]. Simulations ideally would be created automatically from sensor data, experience, and active exploration in the real world. At the moment, however, this is still a manual step. For automated behavior recording, marker free approaches could be tested and compared with respect to accuracy and achievable automation level. Also some prior knowledge is implicitly integrated in the design of the learning platform. There is not one combination of methods that works for all applications. For example, simulation-reality-transfer is only required in challenging applications like ball-throwing.

8 Chapter 8. Automatic Segmentation of Teleoperated Movements

In the previous chapters, it was shown that the developed segmentation algorithms can automatically segment human movement data obtained using motion capture systems such as Qualisys and that the vMCI algorithm can detect meaningful building blocks from which successful robotic behavior can be learned. However, in human-robot interaction, such or similar motion capture systems are not always available. Instead, there are applications in which the human directly controls a robotic system using teleoperation. Teleoperation can be useful in medical environments, handling of nuclear materials, or space and underwater exploration [Lichiardopol, 2007]. Also robotic systems which are able to run autonomously in certain situations may face tasks were their movement repertoire is not sufficient. In these situations, teleoperation is one possibility to execute the desired movements on a system. To teleoperate a system, a haptic interface is needed such as an exoskeleton as proposed in [Mallwitz et al., 2015].

In this chapter, it is shown on several exemplary teleoperated movements that the developed segmentation approaches can also be run on this kind of data. With this, the central movements which are important to fulfill a certain task may be identified during teleoperation. This could, e.g., be used to imitate these movements in order that they are available to the system after teleoperation. An example scenario was developed in the project TransFIT[1], in which a humanoid robotic system is teleoperated

[1] https://robotik.dfki-bremen.de/en/research/projects/transfit/

Figure 8.1.: Recupera Exoskeleton. *Image extracted from [Kumar et al., 2019], Figure 1 (a).*

in a solar panel rotation task. This panel rotation movement should be imitated in order that the system can execute the movement again autonomously. For this, data acquired during teleoperation should be used. A video of this scenario can be found online at: `https://youtu.be/Uwl3XeXvAjo`.

The biggest difference in the automated segmentation of teleoperated movements compared to data acquired using motion tracking is the restriction in movement execution the human operator is confronted with. Whereas the marker-based motion tracking techniques described in chapter 3 do not limit the movement space of the human, an exoskeleton can hinder the operator to move naturally. This is due to kinematic constraints that can be induced by the exoskeleton. These restrictions may result in different movement features compared to naturally performed movements. For example, the velocity of the hand may show different patterns.

In this chapter, it is tested if human movement demonstration obtained during teleoperation can be automatically segmented using vMCI and vHMS, despite the restriction of the movement space of the human. For this, several movements were recorded during teleoperation with an exoskeleton, ranging from a simple one-handed point-to-point movement to a more complex dual arm movement. At the end of this chapter, the results are discussed. The analysis of the point-to-point movement was presented in [Gutzeit et al., 2018b].

8.1. Data Acquisition

Several movements performed with the Recupera exoskeleton [Kumar et al., 2019], which is shown in Figure 8.1, were recorded. The subjects either performed certain movements while wearing the exoskeleton or solved certain teleoperation tasks. During teleoperation, the subjects controlled a simulated version of the RH5 robotic system developed at DFKI RIC[2]. The performed teleoperation movements were recorded by logging the joint positions of the exoskeleton. From these joint positions the Cartesian coordinates of the endeffector and position of the limps were obtained using forward kinematics. Thus, the position of the lower arm, upper arm, and shoulder of the exoskeleton were calculated which roughly correspond to the position of the human wrist, elbow, and shoulder which were also recorded using motion tracking as described in chapter 3. The recorded data was logged at 100 Hz.

8.2. Experiments and Results

Three experiments were performed to test the segmentation approaches on exoskeleton data. First, one-handed point-to-point movement were segmented using vMCI. Afterwards, the hierarchical vHMS segmentation is tested on two dual arm movements, a circle drawing movement, and a dual arm object rotation task. The segmentation algorithms were run on the position and the velocity of the lower arm, whereas the velocity was directly obtained from the position. For this the data was down-sampled to 25 Hz.

8.2.1. Segmentation of point-to-point movements

To obtain a first impression on the applicability of the developed segmentation algorithms on data obtained via an exoskeleton, simple three-dimensional point-to-point movements were recorded from one subject. The subject moved the right arm while wearing the exoskeleton to different positions in space and back. A representative part of the resulting trajectory of the end effector with corresponding velocity can be seen in Figure 8.2. The vMCI algorithm was run on this data to obtain the individual

[2] `https://robotik.dfki-bremen.de/en/research/robot-systems/rh5/`

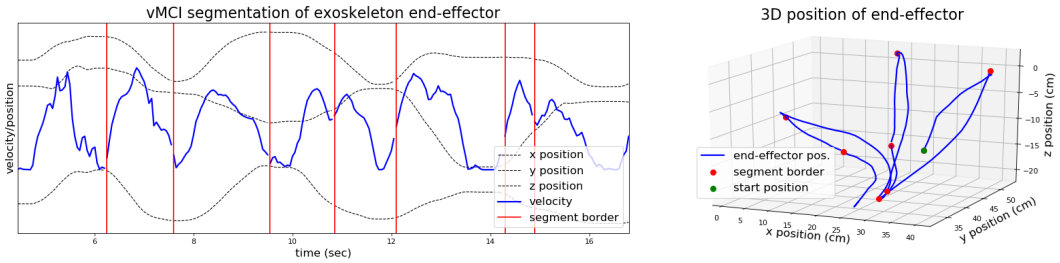


Figure 8.2.: Results obtained by vMCI on point-to-point movements demonstrated by a subject wearing an exoskeleton. The right arm was moved from a rest position to 3 different positions (right side). *Image extracted from [Gutzeit et al., 2018b], Figure 1.*

building blocks movements of the tasks, i.e., the individual point-to-point movements. The resulting segmentation points are visualized in Figure 8.2.

The algorithm successfully found movement segments with a bell-shaped velocity profile in the recorded data. The segments were found without need for data preprocessing and without any parameter tuning. However, some of the movements corresponding to a single point-to-point movement were segmented into two segments instead of one, mainly caused by decelerated movements likely induced by a movement restriction through the exoskeleton.

8.2.2. Segmentation of dual arm circle drawing

In a second experiment, dual arm circle drawing movements were recorded and automatically segmented using the hierarchical segmentation approach vHMS. One subject teleoperated the simulated RH5 system in a way, that the system draws a circle into the air. To segment the resulting end effector position trajectories of the exoskeleton using vHMS, training data needed to be acquired. For this, the same movement performed without the exoskeleton was tracked using the Qualisys motion capture system. This was done to obtain training data at high resolution that is not influenced by possible noise or movement restrictions induced by the exoskeleton. The Qualisys data was manually labeled as described in section 3.8 to generate labeled movement examples for the two dual arm movement actions *circle up* and *circle down*. Additionally, the movements to the starting position (class *move to start*) and the movements of the arms back beside to human body (class *move to rest*) were recorded. The setup for training data recording can be seen in Figure 8.3.

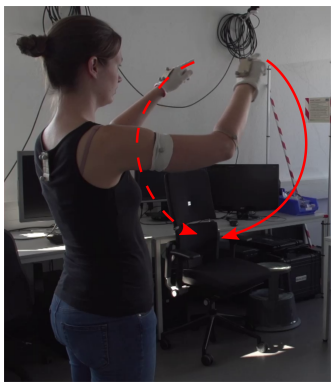


Figure 8.3.: Recording of dual arm movements with action classes *circle up* and *circle down* using the Qualisys system to obtain training data for vHMS segmentation of teleoperated movements.

Using the generated training data, the movement obtained during teleoperation was hierarchically segmented using vHMS. The same training data, containing the classes *move to start*, *circle up*, *circle down*, and *move to rest*, was used for building block detection as well as action detection. The labels were assigned to each individual hand on the building blocks level and to both hands on the action level. This was done because the arms move synchronous in this task. The result of one representative movement demonstration can be seen in Figure 8.4. In the detection of building blocks (Figure 8.4(a)), the data is over-segmented, i.e., several false positive segmentation points were detected which were assigned in the most cases to wrong movement classed. Most likely this results from the noisy velocity profile. However, the dual-arm actions *circle up* and *circle* down were correctly detected in level 1 of the vHMS algorithm, which can be seen in the three-dimensional visualization of the movement in Figure 8.4(b).

8.2.3. Segmentation of dual arm rotation movements

An additional dataset was recorded using the exoskeleton where a subject controlled the simulated RH5 to turn a rectangular frame with both hands. The movements are similar to the movement demonstrations described in section 3.7 and shown in Figure 3.10, but without re-grasping, i.e., a synchronous dual arm movement was performed where both end effectors stay connected to the frame. Due to restrictions in

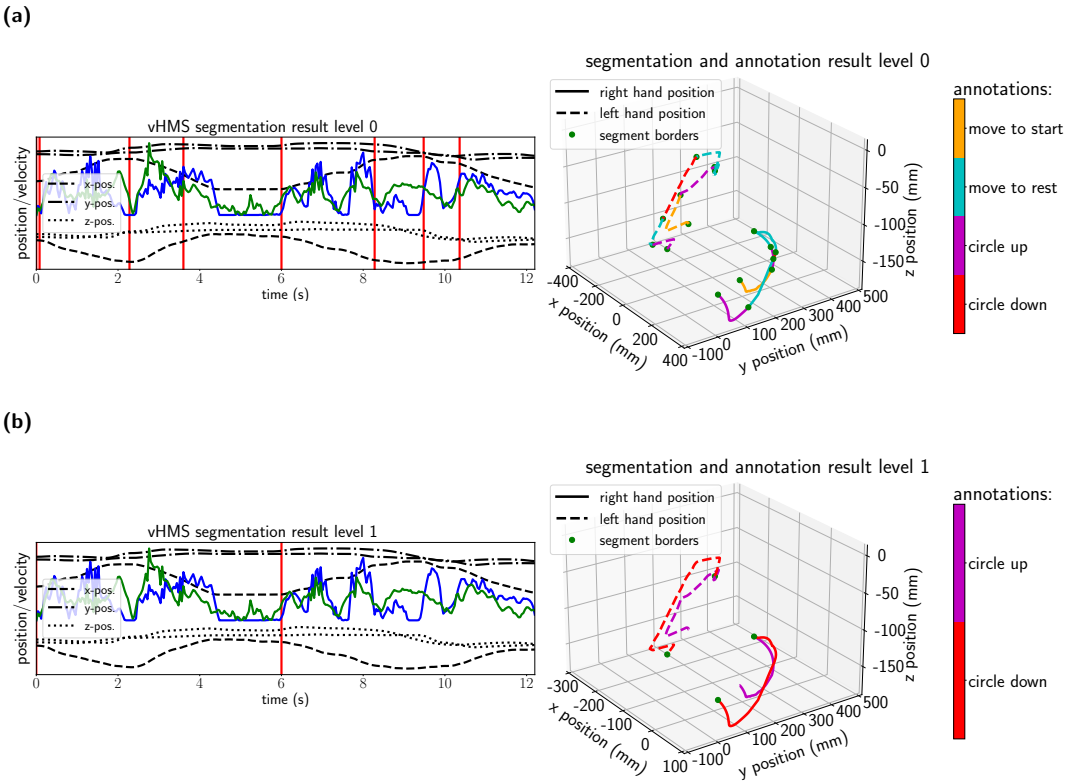


Figure 8.4.: Result of vHMS on circle drawing movements. On the left side, the velocity of both end effectors are shown (blue: right hand, green: left hand) with vertical red lines indicating the detected segment border. On the right side, the three-dimensional movement trajectories are shown, where different colors indicate different detected movement actions. Green dots indicate the detected segment borders. **(a)** shows the intermediate result of vHMS on level 0, **(b)** the final segmentation into movement actions.

the workspace of the exoskeleton and the RH5, the frame was turned by approximately 30 degrees. The recorded data contains several repetitions of the rotation movement consisting of the actions *turn clockwise* and *turn counterclockwise*.

All recorded movements were manually segmented and labeled to obtain a ground truth using the labeling software described in section 3.8. To run vHMS, a training dataset was generated from this manually labeled ground truth, consisting of two movement examples for each of the two actions. The remaining demonstrations (one example per action) were used to test the hierarchical segmentation algorithm vHMS.

Although the segmentation into building blocks showed not that many false pos-

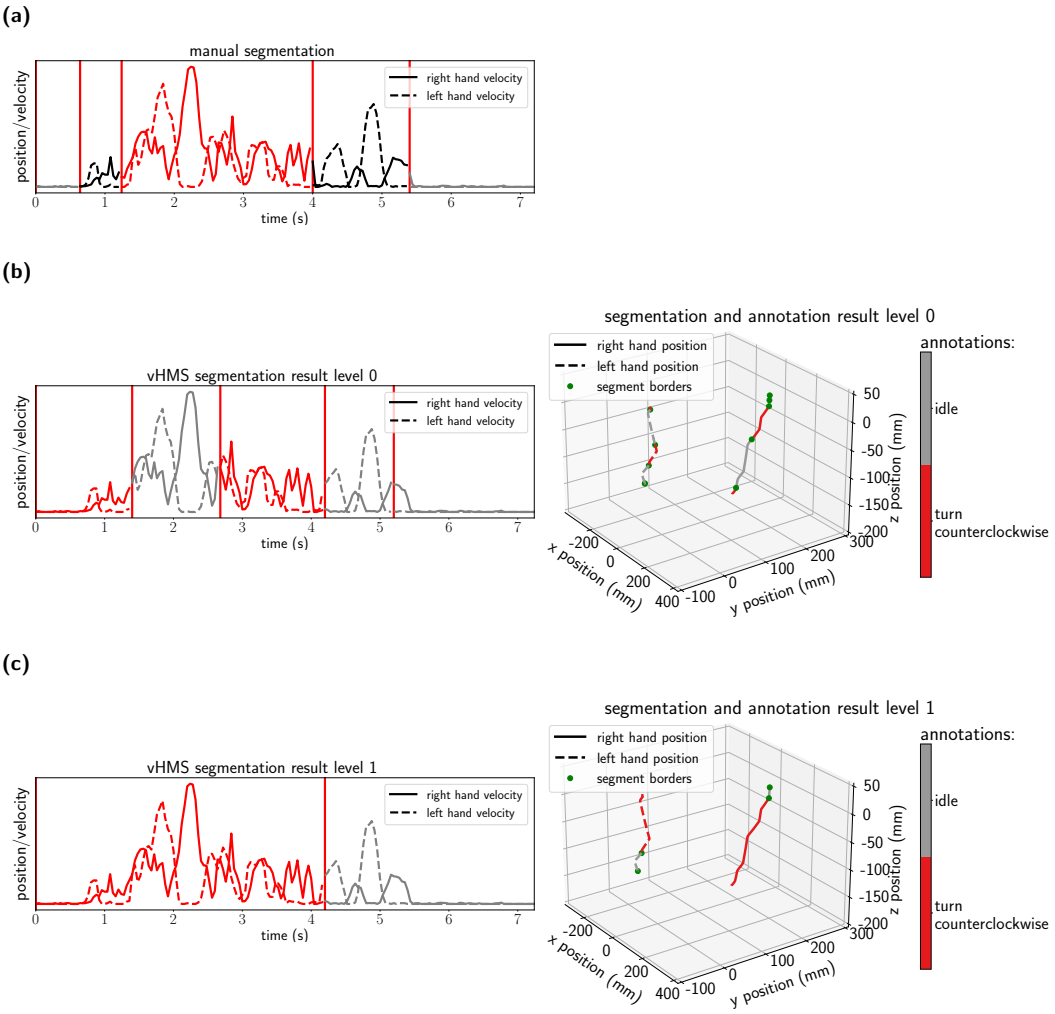


Figure 8.5.: Results of vHMS on teleoperated dual arm rotation movements. Ground truth **(a)**, automatically obtained results on level 0 **(b)**, and final result **(c)** are shown. On the left side, the velocity of both end effectors is shown (blue: right hand, green: left hand), with vertical red lines indicating the detected segment border. On the right side, the three-dimensional movement trajectories are shown, where different colors indicate different detected movement actions. Green dots indicate the detected segment borders.

itives compared to the result of the segmentation of the circle drawing movements presented in the previous section, most parts of the trajectories were not assigned to the correct movement class in this intermediate step of vHMS. This can be seen in the example shown in Figure 8.5(b). This intermediate result corresponds to a vMCI segmentation with successive 1-NN classification which is on this dataset not sufficient to correctly determine the segmentation points and movement classes. The final result of the vHMS segmentation in shown in Figure 8.5(c). In the visualized movement demonstration, the movement class *turn clockwise* was assigned to the correct action label. However, the start of the movement does not correspond to the manually selected start in the ground truth segmentation shown in Figure 8.5(a). The first second of the movement demonstration, which was not assigned to a certain movement class and remained unlabeled in the ground truth, was assigned to the class *turn clockwise* by vHMS. Similar results could be observed if the other movement demonstrations were used as test data.

8.3. Discussion

The results of the experiments described in the previous sections show that the segmentation approaches developed within this thesis can also be used to automatically segment human demonstrations performed while wearing an exoskeleton during teleoperation. For simple point-to-point movements, which were not performed during teleoperation but only by wearing the exoskeleton, the vMCI segmentation method achieved good results and detected single point-to-point movements. In the circle drawing and object rotation task, dual arm movements were recorded during teleoperation of a simulated humanoid robotic system. These movement demonstrations contain more noise in the movement velocity and the movement trajectories are not always direct due to movement constraint origin from the robotic systems. This causes intermediate segments which cannot be mapped to building blocks with a single bell-shaped velocity profile and results in an over-segmentation if the vMCI method is applied. By using the hierarchical vHMS segmentation approach and a small set of manually labeled training data, the performed dual arm actions could be successfully segmented and classified.

However, the presented results only give a first impression of the performance of vHMS on exoskeleton data because of the small number of movement examples that

were evaluated in the performed experiments. Additionally, an evaluation of the algorithms on demonstrations with longer sequences of clearly definable building blocks or actions is needed to deduce a more sophisticated validation of the approaches regarding accuracy in the detection of segment borders.

Part IV.

Conclusion and Outlook

9 Chapter 9. Summary and Conclusion

In this thesis, several methods were developed which automatically analyze human manipulation movements to infer building blocks, their movement labels as well as their combination during the execution of different actions. To evaluate these, several types of human movements were recorded from different subjects such as simple point-to-point movements, throwing movements, as well as more complex dual arm rotation movements. Based on several behavioral and neurobiological studies in the literature and own observations, a bell-shaped velocity profile was identified as a characteristic feature of manipulation building blocks in this thesis. Using this, the vMCI algorithm was developed for unsupervised segmentation of human behavior into building blocks with a bell-shaped velocity profile. To obtain labels for the detected building blocks, a 1-NN classification is suggested, because with 1-NN, building blocks can be classified with a reasonable accuracy by using only a small number of labeled training data. To automatically determine the concatenation of multiple building blocks to movement actions, the vHMS method was developed, in which vMCI and 1-NN are combined into a hierarchical approach that enables to detect building blocks as well as movement actions in one-handed as well as dual arm manipulation movements.

The developed algorithms were applied to learn robotic behavior from human examples as well as to automatically segment data obtained during teleoperation with an exoskeleton. To transfer the detected and labeled building blocks to a robotic system, the BesMan learning platform was developed which is a modular framework for LfD. By using methods for imitation learning, reinforcement learning and transfer learning, identified throwing movements were transferred to different robotic systems

and successful throws were executed. Additionally, it was shown that the vMCI and vHMS algorithm can also be applied to exoskeleton data. In different one-arm and dual arm teleoperated movements, multiple building blocks or movement actions could be detected, despite the higher noise and variance in the movement compared to marker-based motion tracking.

A conclusion for each part of the thesis is given in the following paragraphs. At the end of this chapter, the overall work is concluded before an outlook to future work is given in the next chapter.

Segmentation into Building Blocks In chapter 4, experiments conducted in related publications [Morasso, 1981, Morasso and Mussa-Ivaldi, 1982] were complemented and it was shown that the velocity of the hand is a relevant feature to characterize building blocks in human manipulation movements. The vMCI algorithm was presented in which this knowledge about characteristics of manipulation movements is directly integrated into the segmentation process. VMCI detects building blocks with a bell-shaped profile in the hand velocity in an unsupervised and online manner. The implementation of the algorithm was made public on GitHub. Several experiments were conducted to evaluate vMCI with respect to parameter influence and to compare it to other state-of-the-art methods for unsupervised segmentation. The evaluations were performed on artificial data consisting of two successive DMPs and on real human demonstrations of point-to-point, stick-throwing, and pick-and-place movements, demonstrated by different subjects. The experiments showed that building blocks can reliably be detected using vMCI, also in movement examples with a noisy velocity on which other state-of-the-art segmentation methods decrease in performance. In contrast to these other methods, segmentation points can be determined fast and online, and the algorithm can be applied to different movements without parameter tuning.

Because the bell-shaped velocity is a movement feature that can mainly be observed in manipulation movements, the vMCI approach works best on this kind of data. In other movements, such as gestures or generative movements such as walking, building blocks may show different reoccurring patterns. To detect building blocks in these movements, other approaches may be needed. However, the conducted experiments show that automatic approaches for manipulation movement segmentation can benefit from taking regularities in human movements into account.

Few-shot Recognition To classify building blocks obtained using vMCI, k-NN, HMMs, and LSTMs were compared in chapter 5. Each of the algorithms was evaluated with respect to classification accuracy and time requirements with limited training data on segments detected by vMCI in lever-pulling, pick-and-place, ball-throwing, and stick-throwing movement demonstrations. Additionally, the algorithms were evaluated on gesture data which consist of more complex movements which were manually labeled. The generalization of the classification methods to data of new subjects was evaluated on the stick-throwing and gesture data.

LSTM-based classification gave good results in the recognition of different types of arm movements if the training was performed on very small training set sizes. It also generalized to new subjects in the performed experiments. However, this has to be interpreted with caution, as this is highly dependent on the variations in the examples seen in the training data. If the data is simpler, such as the stick-throwing data analyzed, 1-NN is a clear alternative to LSTM. It requires no hyper-parameter tuning and has faster calculation times on small datasets. Also on the other datasets which were segmented using vMCI, 1-NN showed very good results. While the LSTM network performs better on data with higher inter-subject variations in the experimental evaluation of the generalization abilities, this approach as well as HMM-based classification could not express their superior capabilities on sequenced data in the classification of building blocks of human arm movements.

For the development of embedded multimodal interfaces [Kirchner et al., 2018], simple approaches allow to use miniaturized processing units with relatively low processing power and energy consumption. This is, e.g., relevant in robotics, since the number of interfaces that can be integrated into a robotic system as well as the available computing resources are limited. But also wearable assisting devices have limitations regarding size, energy, and computing power. For these applications not only accurate but also simple methods are needed. With the evaluation performed in chapter 5 it is shown that both, accuracy and simplicity, can be achieved by using vMCI in conjunction with 1-NN classification.

Hierarchical Movement Segmentation In chapter 6, the vHMS algorithm was presented which is a hierarchical movement segmentation method that detects building blocks as well as their concatenation to movement actions in human manipulation movements. Two variants of the algorithm were introduced, one based on classifica-

tion and the other one based on clustering where different methods for classification or clustering can be integrated. vHMS with 1-NN classification to obtain labeled building blocks and movement actions was evaluated in comparison to vHMS with a clustering approach to group building blocks. For the clustering-based variant, k-means and g-means were compared to group the building blocks obtained using vMCI. With both variants, building blocks as well as their concatenations to movement actions could be detected in one-handed point-to-point movements as well as in a dual arm object rotations task.

A hierarchical segmentation can be especially beneficial for data consisting of synchronous and asynchronous dual arm movements, such as the dual arm object rotation dataset. It contains dual arm movements where both hands move synchronously as well as movements where the hands move asynchronously to turn the object. On this data, segmentation points of building blocks were difficult to determine manually to generate a ground truth for evaluation. However, the manual labeling into dual arm movement actions was easier. In the automated detection of building blocks, only 70% of the trajectory points could be assigned to the correct building block using vMCI with g-means clustering. Nonetheless, 80% of the trajectory points could be assigned to the correct dual arm movement action by applying the hierarchical vHMS approach on this complex movement data. Thus, errors in the segmentation into building blocks using vMCI could be corrected during the determination of the movement actions by using vHMS.

Using the presented hierarchical segmentation approach, especially robotic applications which are based on human movement examples can benefit. This is, because not only movement building blocks can be recognized using this method, but also their combination to different movement actions. In this way, e.g., complex movements such as dual arm manipulations could be directly learned from human demonstrations.

Application in robotic Learning from Demonstration In chapter 7, the BesMan learning platform was introduced, which is a framework to learn new robotic behavior from human movement demonstrations. Using the vMCI algorithm with successive 1-NN classification, the building blocks of two different throwing movements could be identified in human demonstrations to use them as movement examples to teach a robotic system to throw. After adapting the movement trajectories in a way that they are executable on the robotic system and optimizing them using reinforcement

or transfer learning, throwing building blocks as well as an approaching and lever-pulling movement could be transferred and successfully executed on a robotic system. This shows that meaningful robotic behavior can be generated from the movement building blocks obtained using vMCI.

Because vMCI is an approach which can be run online and unsupervised and 1-NN needs just a small number of examples for training, most processes needed to learn robotic behavior from human demonstrations could be automated in the learning platform. Furthermore, due to the use of vMCI, building block movements are learned and transferred to the robotic system. Additionally, DMPs are used as movement representation within the learning platform, which are a movement representation that can easily be adapted to slightly changed start or goal positions and which can be optimized for different robotic systems using reinforcement or transfer learning. The recognition of human building blocks in combination with the usage of DMPs as movement representation allows to generate robotic movement building blocks using the presented learning platform.

Segmentation of Teleoperated Movements A further application of the presented segmentation algorithm was given in chapter 8, where movements obtained during teleoperation with an exoskeleton were automatically segmented. In contrast to movement demonstration obtained using marker-based motion tracking, an exoskeleton restricts the demonstrator in its movement execution. Furthermore, the acquired data contains more noise and more movement variations. Nonetheless, it was shown that meaningful segment borders can be detected in the three movement examples acquired with an exoskeleton using vMCI and vHMS. Although a full evaluation on bigger datasets of teleoperated movements is needed to clearly determine the segmentation accuracy of the algorithms on this kind of movements, these first results are promising. Mainly on the dual arm rotation task, in which segment borders are hard to determine even for a human observer, vHMS was able to dissect the data into known actions.

Overall results Several algorithms for automated segmentation and annotation of human manipulation movements were presented in this thesis. The developed algorithms can be used to gain deeper knowledge about observed human movements through the detection and recognition of building blocks in manipulation movements.

As shown in chapter 7, this can directly be used to generate robotic movements by demonstration. Beyond that, the developed methods can be used to gain further insights about movement sequencing in humans, which at the end may help to understand human movement generation better in order that robotic movement may be designed in a similar way. The presented approaches form a basis for new applications in robotics, from which some are discussed in the following chapter.

10 Chapter 10. Outlook

In this chapter, several ideas to improve the presented segmentation and recognition algorithm are presented. The presented approaches can be used to infer building blocks in human movements and their concatenations to movement actions in observed behavior. Future applications of these in LfD applications as well as other human-robot interaction tasks, such as the inference of the intention of the human during a collaboration with a robotic system, will be discussed.

Segmentation and recognition methods There are several ways the segmentation accuracy of vMCI may be improved. Currently, no preprocessing of the data is required in order to run vMCI except for a down-sampling. Without down-sampling, vMCI tends to over-segment the data. To circumvent this, the influence of the algorithms' hyper-parameter on the segmentation accuracy should be evaluated with respect to different frequencies of the data. Possibly, the over-segmentation at high frequencies could be improved by initializing some hyper-parameter, such as the expected segment length p in relation to the frequency of the data. Additionally, vMCI detects segmentation points only correctly within a margin around the true velocity minima. This could be improved by combining the approach with a method that detects local minima exactly, such as the locMin approach vMCI is compared to in the performed experiments. By combining the determination of exact segmentation points of locMin with the generally applicable vMCI approach which does not require hyper-parameter adaption for different datasets, segmentation accuracy could be improved.

In vMCI, segment borders are determined at positions where the underlying auto-

regressive LRM for the marker positions change and bell-shaped velocity curve ends. To account for superimposed movement building blocks which result in overlapping velocity curves, as shown in Figure 2.5, different positions of the velocity peaks and different start and end velocities are already modeled within vMCI. An alternative would be to adapt the algorithm to an approach which internally uses building blocks with a bell-shaped velocity curve and directly models their overlap. This could be done by assuming that the velocity is modeled as a weighted sum of single radial basis functions to model also overlapping building blocks. By determining points where single basis functions overlap, the segmentation points may be detected. A comparison with the proposed vMCI approach would be of interest, not only in terms of segmentation accuracy but also with respect to computation time.

To annotate building blocks detected using vMCI, good results were achieved by using simple 1-NN classification, possibly because vMCI reduced the complexity of the data. For future work, a more detailed analysis of this influence of vMCI segmentation on the recognition accuracy of 1-NN would be of interest, also in comparison to other movement recognition approaches. Additionally, a combined approach of movement segmentation and recognition, in which not only the building blocks but also their labels are detected online should be developed. From this, especially applications in human-robot collaboration, where the current behavior should be understood by the system, might benefit. With an online detection and recognition of building blocks, also the vHMS algorithm for automated detection of labeled building blocks and their combination to movement actions can be improved. Due to the computational complexity of vHMS, which is exponential with respect to the number of detected building blocks, the method works best on movement demonstrations consisting of a limited number of building blocks. If building blocks and their labels can be determined online, also the vHMS approach can be extended to an online approach which is less computationally expensive and might be run in long-term applications to infer the current building blocks and actions of the human.

Applications of proposed approaches In part III of this thesis, already two applications of the presented segmentation and recognition methods were proposed. Especially for robot learning from demonstration, the detection of building blocks in human movements could be important to learn basic and generally applicable robotic movements. For this, the presented approaches to detect building blocks in human

movements which can serve as movement examples for the generation of robotic behavior are just the first step. In future work, it should be evaluated if the robotic movements learned based on the human examples can be used and recombined to generate different behaviors. If this is the case, robots may solve complex manipulation tasks similar to humans by sequentially learning individual building blocks and combining them to solve different goals. This probably will be more effective in terms of computational resources compared to monolithic approaches. Thus, there would be more resources left for other computations. To achieve this, one of the next steps based on the work presented within this thesis is the usage of the hierarchical segmentation to hierarchically learn robotic movements based on human examples. A comparison of this to other hierarchical approaches, which learn new behavior from scratch without human demonstration, such as approaches based on hierarchical reinforcement learning, would be of interest.

Furthermore, other applications in human-robot interaction might benefit from the proposed approaches. As described in section 1.1, a detailed knowledge of the current human behavior as well as the intentions of the human are needed to realize a best possible assistance of a system during a collaboration with a human. For this, the developed algorithms can be used to determine the currently observed building block and infer the next most likely building blocks the human may execute. On this topic, there is already conducted research in the project HaLeR[1], where new methods for movement prediction are developed based on the methods presented in this thesis.

The methods presented in this thesis are restricted to the pure analyses of the movement trajectories, without considering other sensory input, such as gaze, body postures or changes in the environment, resulting, e.g., from changing positions of the object which should be manipulated. However, as described in chapter 2, human movement generation is not restricted to the execution of a desired trajectory. Instead, motor command generation, state transition, and the generation of sensory feedback are closely connected to generate the movements required to fulfill certain tasks. If a robotic system should react as intuitively as humans in different situations despite a continuously changing environment, more sensory input should be taken into account. This is not only important on the level of behavior analysis as treated in this thesis but also during movement execution. Integrating all these different aspects of human-like generation of robotic behavior is one of the big challenges in the future.

[1] https://robotik.dfki-bremen.de/en/research/projects/haler/

Part V.

Appendix

A Appendix A. Derivation of the model evidence

A.1. Calculation of the model evidence in vMCI

The likelihood of the observed data sequence $p(y_{i+1:j}|m, m_v)$ given the model m of order q and velocity model m_v, also called model evidence, will be derived in this section. By marginalizing out the model parameters β and α as given in equation (4.4) the model evidence can be solved directly if conjugate priors are chosen for the LRMs the data is modeled with. To derive the result given in equation (4.5), the likelihood of the position data $p(y^p_{i+1:j}|m)$ is derived first.

A.1.1. Calculation of $p(y^p_{i+1:j}|m)$

The likelihood of the position data, $p(y^p_{i+1:j}|m)$, is defined as

$$p(y^p_{i+1:j}|m) = \int p(y^p_{i+1:j}|\beta, m)p(\beta)\, d\beta. \tag{A.1}$$

In this equation, the parameters β are assumed to be matrix normal distributed with zero mean and covariances D and Σ, i.e.,

$$\beta \sim \mathcal{MN}(0, D, \Sigma). \tag{A.2}$$

As a conjugate prior, the covariance Σ is inverse Wishart distributed, i.e.,

$$\Sigma \sim \mathcal{IW}(\nu, S). \tag{A.3}$$

With these prior distributions, the parameters β in (A.1) can be marginalized out, which leads to the following equation (which is identical to the first part in equation (4.4)):

$$p(y^p_{i+1:j}|m) = \int\int p(y^p_{i+1:j}|\beta,\Sigma)\cdot p(\beta|D,\Sigma)\cdot p(\Sigma|\nu,S)\,d\Sigma\,d\beta. \tag{A.4}$$

To simplify, y is written instead of $y^p_{i+1:j}$ during the determination of the model evidence and solve the integrals in (A.4) using the precision $\Lambda = \Sigma^{-1}$, which leads to

$$\begin{aligned} p(y|m) &= \int\int p(y|\beta,\Lambda)\cdot p(\beta|D,\Lambda)\cdot p(\Lambda|\nu,S)\,d\Lambda\,d\beta \\ p(y|m) &= \int p(\Lambda|\nu,S)\left(\int p(y|\beta,\Lambda)\cdot p(\beta|D,\Lambda)\cdot d\beta\right) d\Lambda. \end{aligned} \tag{A.5}$$

In order to determine the model evidence, the probability density functions of the matrix normal distribution and the Wishart distribution is needed. A random matrix $X(n\times d)$ that is **matrix normal distributed**, i.e., $X \sim \mathcal{MN}(M_X, V, W)$, with $M_X(n\times d)$, $V(n\times n)$ and $W(d\times d)$, has the probability density function:

$$P(X) = (2\pi)^{-\frac{nd}{2}}|V|^{-\frac{d}{2}}|W|^{-\frac{n}{2}}\exp\left\{-\frac{1}{2}tr\left((X-M_X)^TV^{-1}(X-M_X)W^{-1}\right)\right\}, \tag{A.6}$$

which can be reformulated to:

$$P(X) = (2\pi)^{-\frac{nd}{2}}|V|^{-\frac{d}{2}}|W|^{-\frac{n}{2}}\exp\left\{-\frac{1}{2}tr\left((X^TV^{-1}X - X^TV^{-1}M_X)W^{-1}\right)\right\}. \tag{A.7}$$

And a random matrix $A(d\times d)$ that is **Wishart distributed**, i.e., $A \sim \mathcal{W}(\nu, S^{-1})$, with scalar ν and $S(d\times d)$, has the probability density function:

$$P(A) = \frac{|S|^{\frac{\nu}{2}}}{\Gamma_d(\frac{\nu}{2})2^{\frac{\nu d}{2}}}|A|^{\frac{\nu-d-1}{2}}\exp\left\{-\frac{1}{2}tr\,(SA)\right\}, \tag{A.8}$$

where $\Gamma_d(n)$ is the multivariate Gamma function defined as:

$$\Gamma_d(n) = \int \exp\{-tr(S)\}|S|^{n-\frac{p+1}{2}}. \tag{A.9}$$

Using these definitions, the integral in (A.5) can be solved step by step as follows:

1. determination of the posterior probability of the model parameters $p(\beta|y,D,\Lambda)$, which is due to Bayes rule proportional to the first two factors in the integral in equation (A.5), i.e.: $p(\beta|y,D,\Lambda) \propto p(y|\beta,\Lambda)\cdot p(\beta|D,\Lambda)$
2. solving the integral over parameters β in (A.5)
3. solving the integral over precision Λ in (A.5)

All these steps are described in detail in the following paragraphs.

1. Determination of posterior probability $p(\beta|y, D, \Lambda)$: From the representation of the data sequence as a linear regression model as defined in (4.1), follows that the observed data is matrix-normal distributed with mean $H\beta$ and covariances I and Σ, with H defines as the matrix of basis functions, i.e., $H = (\phi_1, ..., \phi_q)$ of shape $n \times q$, and I is the identity matrix. By using this, the distribution of β given in (A.2), the probability density function of the matrix normal distribution and by using the precision Λ instead of Σ, the posterior probability of β can be derived as follows:

$$\begin{aligned}
p(\beta|y, D, \Lambda) &\propto p(y|\beta, \Lambda) \cdot p(\beta|D, \Lambda) \\
&= \mathcal{MN}(y|H\beta, I, \Lambda) \cdot \mathcal{MN}(\beta|D, \Lambda) \\
&= (2\pi)^{-\frac{nd}{2}} |I|^{-\frac{d}{2}} |\Lambda|^{\frac{n}{2}} \exp\left\{-\frac{1}{2} tr\left((y - H\beta)^T I^{-1}(y - H\beta)\Lambda\right)\right\} \\
&\quad \cdot (2\pi)^{-\frac{qd}{2}} |D|^{-\frac{d}{2}} |\Lambda|^{\frac{q}{2}} \exp\left\{-\frac{1}{2} tr\left(\beta^T D^{-1}\beta\Lambda\right)\right\} \\
&= (2\pi)^{-\frac{nd}{2}-\frac{qd}{2}} |D|^{-\frac{d}{2}} |\Lambda|^{\frac{n}{2}+\frac{q}{2}} \\
&\quad \cdot \exp\underbrace{\left\{-\frac{1}{2} tr\left[\left((y - H\beta)^T(y - H\beta) + \beta^T D^{-1}\beta\right)\Lambda\right]\right\}}_{\star}. \qquad \text{(A.10)}
\end{aligned}$$

Because a conjugate Gaussian prior is used to model the data, the posterior of the model parameter will again be matrix normal distributed. From this follows, that the exponent $\star$ can be reformulated to derive the mean and covariances of the posterior distribution. As suggested in [Bishop, 2006], a technique called "completing the square" is applied by reformulating and sorting $\star$ to derive all summands that are squared in β, linear in β and constant in order that $\star$ can be mapped to the matrix normal distribution given in (A.7):

$$\begin{aligned}
\star &= \left((y - H\beta)^T(y - H\beta) + \beta^T D^{-1}\beta\right)\Lambda \\
&= \left(y^T y - y^T H\beta - (H\beta)^T y + (H\beta)^T H\beta + \beta^T D^{-1}\beta\right)\Lambda \\
&= \left((H\beta)^T H\beta + \beta^T D^{-1}\beta - y^T H\beta - (H\beta)^T y + y^T y\right)\Lambda \\
&= \left(\beta^T H^T H\beta + \beta^T D^{-1}\beta - \beta^T H^T y - \beta^T H^T y + y^T y\right)\Lambda \\
&= \left(\beta^T (H^T H + D^{-1})\beta - 2\beta^T H^T y + y^T y\right)\Lambda.
\end{aligned}$$

The first summand of $\star$ has now the same form as in equation (A.7). Thus, the covariance of the matrix normal distribution of β ca be defined as:

$$M = (H^T H + D^{-1})^{-1}. \tag{A.11}$$

To derive the mean μ, the second summand in $\star$ is needed, which should according to equation (A.7) be equal to $\beta^T M^{-1} \mu$ leading to:

$$\begin{aligned} & \beta^T M^{-1} \mu && = \beta^T H^T y \\ \Leftrightarrow\ & M^{-1} \mu && = H^T y \\ \Leftrightarrow\ & \mu && = M H^T y. \end{aligned}$$

By inserting these definitions of M and μ in(A.10) one gets:

$$\begin{aligned} p(y|\beta, \Lambda) \cdot p(\beta|D, \Lambda) = (2\pi)^{-\frac{nd}{2} - \frac{qd}{2}} |D|^{-\frac{d}{2}} |\Lambda|^{\frac{n}{2} + \frac{q}{2}} \\ \cdot \exp\left\{ -\frac{1}{2} tr\left[\left(\beta^T M^{-1} \beta - 2\beta^T M^{-1} \mu + y^T y \right) \Lambda \right] \right\} \end{aligned} \tag{A.12}$$

2. Solving integral over β: By using equation (A.12), the integral over β in equation (A.5) can be solved:

$$\begin{aligned}
\int p(y|\beta,\Lambda)p(\beta|D,\Lambda)\,d\beta &= \int (2\pi)^{-\frac{nd}{2}-\frac{qd}{2}}|D|^{-\frac{d}{2}}|\Lambda|^{\frac{n}{2}+\frac{q}{2}}\cdot\exp\left\{-\frac{1}{2}tr\left[\left(\beta^T M^{-1}\beta\right.\right.\right.\\
&\qquad \left.\left.\left.-2\beta^T M^{-1}\mu+y^T y\right)\Lambda\right]\right\}\,d\beta\\
&= (2\pi)^{-\frac{nd}{2}-\frac{qd}{2}}|D|^{-\frac{d}{2}}|\Lambda|^{\frac{n}{2}+\frac{q}{2}}\cdot\int\exp\left\{-\frac{1}{2}tr\left[\left(\beta^T M^{-1}\beta\right.\right.\right.\\
&\qquad \left.\left.\left.-2\beta^T M^{-1}\mu+\mu^T M^{-1}\mu-\mu^T M^{-1}\mu+y^T y\right)\Lambda\right]\right\}\,d\beta\\
&= (2\pi)^{-\frac{nd}{2}-\frac{qd}{2}}|D|^{-\frac{d}{2}}|\Lambda|^{\frac{n}{2}+\frac{q}{2}}\cdot\int\exp\left\{-\frac{1}{2}tr\left[\left((\beta-\mu)^T M^{-1}\right.\right.\right.\\
&\qquad \left.\left.\left.\cdot(\beta-\mu)-\mu^T M^{-1}\mu+y^T y\right)\Lambda\right]\right\}\,d\beta\\
&= (2\pi)^{-\frac{nd}{2}-\frac{qd}{2}}|D|^{-\frac{d}{2}}|\Lambda|^{\frac{n}{2}+\frac{q}{2}}\cdot\exp\left\{-\frac{1}{2}tr\left[\left(-\mu^T M^{-1}\mu+y^T y\right)\Lambda\right]\right\}\\
&\quad\cdot\int\exp\left\{-\frac{1}{2}tr\left[\left((\beta-\mu)^T M^{-1}(\beta-\mu)\right)\Lambda\right]\right\}\,d\beta\\
&= (2\pi)^{-\frac{nd}{2}-\frac{qd}{2}}|D|^{-\frac{d}{2}}|\Lambda|^{\frac{n}{2}+\frac{q}{2}}\cdot\exp\left\{-\frac{1}{2}tr\left[\underbrace{\left(-\mu^T M^{-1}\mu+y^T y\right)}_{\hat{B}}\Lambda\right]\right\}\\
&\quad\cdot(2\pi)^{\frac{qd}{2}}|\Lambda|^{-\frac{q}{2}}|M|^{\frac{d}{2}}\\
&= (2\pi)^{-\frac{nd}{2}}|D|^{-\frac{d}{2}}|\Lambda|^{\frac{n}{2}}|M|^{\frac{d}{2}}\exp\left\{-\frac{1}{2}tr\left[\hat{B}\Lambda\right]\right\} \qquad \text{(A.13)}
\end{aligned}$$

3. Solving integral over Λ: Due to the chosen conjugate priors, Σ is inverse Wishart distributed as defined in equation (A.3). From this follows that Λ, which is defined as the inverse of Σ is Wishart distributed, $\Lambda \sim \mathcal{W}(\nu, S^{-1})$, with probability density function given in (A.8). Using this and the solution of the integral over β given in

equation (A.13), the integral over Λ in equation (A.5) ca be solved:

$$\begin{aligned}
p(y|m) &= \int p(\Lambda|\nu, S) \left(\int p(y|\beta, \Lambda) \cdot p(\beta|D, \Lambda) \cdot d\beta \right) d\Lambda \\
&= \int \frac{|S|^{\frac{\nu}{2}}}{\Gamma_d(\frac{\nu}{2}) 2^{\frac{\nu d}{2}}} |\Lambda|^{\frac{\nu-d-1}{2}} \exp\left\{ -\frac{1}{2} tr\left(S\Lambda\right) \right\} \\
&\qquad \cdot (2\pi)^{-\frac{nd}{2}} |D|^{-\frac{d}{2}} |\Lambda|^{\frac{n}{2}} |M|^{\frac{d}{2}} \exp\left\{ -\frac{1}{2} tr\left[\hat{B}\Lambda\right] \right\} d\Lambda \\
&= (2\pi)^{-\frac{nd}{2}} |D|^{-\frac{d}{2}} |M|^{\frac{d}{2}} |S|^{\frac{\nu}{2}} \frac{1}{\Gamma_d(\frac{\nu}{2}) 2^{\frac{\nu d}{2}}} \\
&\qquad \cdot \int |\Lambda|^{\frac{n+\nu}{2} - \frac{d+1}{2}} \exp\left\{ -\frac{1}{2} tr\left(\underbrace{(S + \hat{B})}_{B} \Lambda \right) \right\} d\Lambda. \qquad \text{(A.14)}
\end{aligned}$$

To solve the integral in (A.14), $B\Lambda$ is substituted by Z, i.e. $Z = B\Lambda \Rightarrow \Lambda = B^{-1}Z$ and $|\Lambda| = |B^{-1}||Z|$. By using the definition of the Gamma function given in (A.9) which leads to:

$$\begin{aligned}
p(y|m) &= (2\pi)^{-\frac{nd}{2}} |D|^{-\frac{d}{2}} |M|^{\frac{d}{2}} |S|^{\frac{\nu}{2}} \frac{1}{\Gamma_d(\frac{\nu}{2}) 2^{\frac{\nu d}{2}}} \\
&\qquad \cdot \int |B|^{-\frac{n+\nu}{2} + \frac{d+1}{2}} |Z|^{\frac{n+\nu}{2} - \frac{d+1}{2}} \exp\left\{ -\frac{1}{2} tr\left(Z\right) \right\} |B|^{\frac{d+1}{2}} dZ \\
&= (2\pi)^{-\frac{nd}{2}} |D|^{-\frac{d}{2}} |M|^{\frac{d}{2}} |S|^{\frac{\nu}{2}} \frac{1}{\Gamma_d(\frac{\nu}{2}) 2^{\frac{\nu d}{2}}} |B|^{-\frac{n+\nu}{2}} \Gamma_d(\frac{n+\nu}{2}) \\
&= (2\pi)^{-\frac{nd}{2}} \frac{|M|^{\frac{d}{2}}}{|D|^{\frac{d}{2}}} \frac{|S|^{\frac{\nu}{2}}}{|B|^{\frac{n+\nu}{2}}} \frac{\Gamma_d(\frac{n+\nu}{2})}{\Gamma_d(\frac{\nu}{2}) 2^{\frac{\nu d}{2}}} \qquad \text{(A.15)}
\end{aligned}$$

With $M = (H^T H + D^{-1})^{-1}$ and $\mu = MH^t y$, B simplifies to:

$$\begin{aligned}
B &= S + \hat{B} \\
&= S + y^T y - \mu^T M^{-1} \mu \\
&= S + y^t y - y^t (MH^T)^T M^{-1} MH^T y \\
&= S + y^t y - y^t HMH^t Y \\
&= S + y^t \underbrace{(I - HMH^t)}_{P} y.
\end{aligned}$$

With this, the model evidence for the position given in (A.15) is equal to:

$$p(y|m) = (2\pi)^{-\frac{nd}{2}} \frac{|M|^{\frac{d}{2}}}{|D|^{\frac{d}{2}}} \frac{|S|^{\frac{\nu}{2}}}{|y^T P y + S|^{\frac{n+\nu}{2}}} \frac{\Gamma_d(\frac{n+\nu}{2})}{\Gamma_d(\frac{\nu}{2}) 2^{\frac{\nu d}{2}}}, \tag{A.16}$$

A.1.2. Calculation of $p(y_{i+1:j}|m, m_v)$

In the previous section, the calculation of the likelihood of the position data given model m, $p(y^p_{i+1:j}|m)$, was derived. The result is given in equation (A.16). Because the velocity data $y^v_{i+1:j}$ is also modeled with the linear regression model m_v and conjugate prior, see (4.3), the model evidence $p(y^v_{i+1:j}|m_v)$ can be derived in the same way as $p(y^p_{i+1:j}|m)$ leading to:

$$p(y^v_{i+1:j}|m_v) = (2\pi)^{-\frac{nd_v}{2}} \frac{|M_v|^{\frac{d_v}{2}}}{|D_v|^{\frac{d_v}{2}}} \frac{|S_v|^{\frac{\nu_v}{2}}}{|(y^v)^T P_v y^v + S_v|^{\frac{n+\nu_v}{2}}} \frac{\Gamma_{d_v}(\frac{n+\nu_v}{2})}{\Gamma_{d_v}(\frac{\nu_v}{2}) 2^{\nu_v d_v/2}} \quad . \tag{A.17}$$

With (A.16) and (A.17) the model evidence of the whole data $y_{i+1:j}$ becomes:

$$\begin{aligned} p(y_{i+1:j}|m, m^v) = {} & (2\pi)^{-\frac{nd}{2}} \frac{|M|^{\frac{d}{2}}}{|D|^{\frac{d}{2}}} \frac{|S|^{\frac{\nu}{2}}}{|(y^p)^T P y^p + S|^{\frac{n+\nu}{2}}} \frac{\Gamma_d(\frac{n+\nu}{2})}{\Gamma_d(\frac{\nu}{2}) 2^{\nu d/2}} \\ & \cdot (2\pi)^{-\frac{nd_v}{2}} \frac{|M_v|^{\frac{d_v}{2}}}{|D_v|^{\frac{d_v}{2}}} \frac{|S_v|^{\frac{\nu_v}{2}}}{|(y^v)^T P_v y^v + S_v|^{\frac{n+\nu_v}{2}}} \frac{\Gamma_{d_v}(\frac{n+\nu_v}{2})}{\Gamma_{d_v}(\frac{\nu_v}{2}) 2^{\nu_v d_v/2}} \quad , \end{aligned}$$

as written down in equation (4.5).

B Appendix B.

Contribution to Publications

Most parts of thesis were already published in international journals, bookchapters or international and national conferences. In this section, all publications are listed including a description of my contributions. The listed publications are sorted by date in ascending order. Please note that earlier publications were published under the name Lisa Senger.

B.1. International Journals and Bookchapters

[Metzen et al., 2013] Metzen, J. H., Fabisch, A., **Senger, L.**, Gea Fernández, J., and Kirchner, E. A. "Towards learning of generic skills for robotic manipulation". *KI - Künstliche Intelligenz*, 28(1), pages 15–20, 2013.

- Contributions of Lisa Gutzeit: Contributed to the description of the BesMan learning platform with focus on the "Behavior Segmentation" module and reviewed the paper.

[Gutzeit et al., 2016] **Gutzeit, L.**, Otto, M., and Kirchner, E. A. "Simple and robust automatic detection and recognition of human movement patterns in tasks of different complexity". *In Physiological Computing Systems,* pages 39–57, Springer, 2016.

- Contributions of Lisa Gutzeit: Wrote most parts of the paper, including introduction, related work, description of methods, and experiments. Implemented the approaches for movement segmentation and recognition. Recorded or assisted

in the recording of the movement data, evaluated the presented approaches on two of the datasets and assisted in the evaluation on the third dataset.

[Gutzeit et al., 2018a] **Gutzeit, L.**, Fabisch, A., Otto, M., Metzen, J. H., Hansen, J., Kirchner, F., and Kirchner, E. A. "The BesMan Learning Platform for Automated Robot Skill Learning". *Frontiers in Robotics and AI*, 5, 2018.

- Contributions of Lisa Gutzeit: Wrote sections about data acquisition, movement segmentation, application of the learning platform in different scenarios and parts of the introduction, methods, and discussion sections. Wrote ethics proposal, presented it to ethics commission, and organized the data acquisition. Implemented and evaluated the methods for behavior segmentation and recognition.

[Gutzeit and Kirchner, 2022] **Gutzeit, L.** and Kirchner, F. "Unsupervised Segmentation of Human Manipulation Movements into Building Blocks". *IEEE Access*, 2022.

- Contributions of Lisa Gutzeit: Wrote and revised most parts of the paper. Implemented and evaluated segmentation approach. Recorded the movement data.

B.2. International Conferences

[Senger et al., 2014] **Senger, L.**, Schröer, M., Metzen, J. H., and Kirchner, E. A. "Velocity-based Multiple Change-point Inference for Unsupervised Segmentation of Human Movement Behavior". *In Proceedings of the 22th International Conference on Pattern Recognition (ICPR2014)*, pages 4564–4569, 2014.

- Contributions of Lisa Gutzeit: Wrote and revised most parts of the paper. Implemented and evaluated segmentation approach.

[Gutzeit and Kirchner, 2016] **Gutzeit, L**. and Kirchner, E. A. "Automatic detection and recognition of human movement patterns in manipulation tasks." *In Proceedings of the 3rd International Conference on Physiological Computing Systems.*, 2016.

- Contributions of Lisa Gutzeit: Wrote and revised most parts of the paper. Implemented and evaluated segmentation approach. Recorded or assisted in the recording of the movement data.

[Gutzeit et al., 2018b] (abstract + poster) **Gutzeit, L.**, Tabie, M., and Kirchner, E. A. "Automatic movement segmentation of exoskeleton data'". *In Conference Proceedings of the 3rd International Mobile Brain/Body Imaging Conference, (MoBI-2018)*, 2018.

- Contributions of Lisa Gutzeit: Wrote parts of the abstract. Implemented and evaluated the segmentation of the exoskeleton-data.

[Gutzeit, 2021] **Gutzeit, L.**. "A Comparison of Few-shot Classification of Human Movement Trajectories". *In Proceedings of the 10th International Conference on Pattern Recognition Applications and Methods - Volume 1: ICPRAM*, pages 243–250, INSTICC, SciTePress, 2021.

- Contributions of Lisa Gutzeit: wrote whole paper, recorded the movement data or assisted in the recording, implemented methods, and evaluated the results.

[Gutzeit, 2022] **Gutzeit, L.**. "Hierarchical Segmentation of Human Manipulation Movements". *In Proceedings of the 26th International Conference on Pattern Recognition (ICPR2022)*, 2022.

- Contributions of Lisa Gutzeit: Wrote whole paper, recorded the movement data, implemented methods, and evaluated the results.

B.3. National Conferences

[Schreiter et al., 2014] Schreiter, L., **Senger, L.**, Beyl, T., Berghöfer, E., Raczkowsky, J., and Woern, H. "Probabilistische Echtzeit-Situationserkennung im Operationssaal am Beispiel von OP:Sense". *In Tagungsband der 13. Jahrestagung der Deutschen Gesellschaft für Computer- und Roboterassistierte Chirurgie e.V., (CURAC-2014),* pages 177–180, 2014.

- Contributions of Lisa Gutzeit: Contributed to the implementation of the movement recognition and revised the paper.

[Schreiter et al., 2015] Schreiter, L., Berghöfer, E., Beyl, T., **Senger, L.**, Raczkowsky, J., Kirchner, F., and Woern, H. "Probabilistic Situation Detection for Human- Robot Interaction in an OP Lab Environment". *In Tagungsband der 14. Jahrestagung der Deutschen Gesellschaft für Computer- und Roboterassistierte Chirurgie e.V, (CURAC-15)*, pages 99–104, 2015.

- Contributions of Lisa Gutzeit: Contributed to the implementation of the movement recognition and revised the paper.

[Gutzeit et al., 2019] **Gutzeit, L.**, Fabisch, A., Petzoldt, C., Wiese, H., and Kirchner, F. "Automated Robot Skill Learning from Demonstration for Various Robot Systems". *In Benzmüller, C. and Stuckenschmidt, H., editors, KI 2019: Advances in Artificial Intelligence, Conference Proc., volume LNAI 11793,* pages 168–181, Springer, 2019.

- Contributions of Lisa Gutzeit: Wrote section about movement segmentation and parts of introduction, experiments, results, and conclusion. Contributed to the development of evaluation approaches for the learning platform and assisted in recording of the movement data.

Appendix C.

C Abbreviations

1-NN 1-Nearest Neighbor

BPARHMM Beta-process auto-regressive Hidden Markov Model

CNN Convolutional Neural Network

CNS central nervous system

DMP Dynamical Movement Primitive

DTW Dynamic Time Warping

FP false positives

HMM Hidden Markov Model

IMU inertial measurement unit

k-NN k-Nearest Neighbor

LfD Learning from Demonstration

locMin local minima segmentation

LRM linear regression model

LSTM Long-short Term Memory-Network

MCI Multiple Change-point Inference

PCA Principal Component Analysis

ProbS Probabilistic Segmentation

RNN Recurrent Neural Network

TP true positives

t-SNE t-distributed Stochastic Neighbor Embedding

vHMS velocity-based Hierarchical Movement Segmentation

vMCI velocity-based Multiple Change-point Inference

Bibliography

[Aarno and Kragic, 2008] Aarno, D. and Kragic, D. (2008). Motion intention recognition in robot assisted applications. *Robotics and Autonomous Systems*, 56:692–705.

[Adi-Japha et al., 2008] Adi-Japha, E., Karni, A., Parnes, A., Loewenschuss, I., and Vakil, E. (2008). A shift in task routines during the learning of a motor skill: Group-averaged data may mask critical phases in the individuals' acquisition of skilled performance. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 24:1544–1551.

[Aggarwal and Cai, 1999] Aggarwal, J. and Cai, Q. (1999). Human motion analysis: A review. *Computer Vision and Image Understanding*, 73(3):428–440.

[Agostini et al., 2020] Agostini, A., Saveriano, M., Lee, D., and Piater, J. (2020). Manipulation Planning Using Object-Centered Predicates and Hierarchical Decomposition of Contextual Actions. *IEEE Robotics and Automation Letters*, 5(4):5629–5636.

[Andrychowicz et al., 2020] Andrychowicz, O. A. M., Baker, B., Chociej, M., Józefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., Schneider, J., Sidor, S., Tobin, J., Welinder, P., Weng, L., and Zaremba, W. (2020). Learning dexterous in-hand manipulation. *International Journal of Robotics Research*, 39(1):3–20.

[Argall et al., 2009] Argall, B. D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483.

[Bargsten and de Gea Fernández, 2015] Bargsten, V. and de Gea Fernández, J. (2015). Compi: Development of a 6-dof compliant robot arm for human-robot cooperation. In *Proceedings of the 8th International Workshop on Human-Friendly Robotics*. Technische Universität München (TUM).

[Bartsch et al., 2016] Bartsch, S., Manz, M., Kampmann, P., Dettmann, A., Hanff, H., Langosz, M., v. Szadkowski, K., Hilljegerdes, J., Simnofske, M., Kloss, P., Meder, M., and Kirchner, F. (2016). Development and control of the multi-legged robot mantis. In *Proceedings of ISR 2016: 47st International Symposium on Robotics*, pages 1–8.

[Beers et al., 1999] Beers, R. J. V., Sittig, A. C., and Gon, J. J. D. V. D. (1999). Integration of proprioceptive and visual position-information: An experimentally supported model. *Journal of Neurophysiology*, 81.

[Berthier et al., 2005] Berthier, N. E., Rosenstein, M. T., and Barto, A. G. (2005). Approximate optimal control as a model for motor learning. *Psychological Review*, 112:329–346.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc.

[Bizzi et al., 1991] Bizzi, E., Mussa-Ivaldi, F., and Giszter, S. (1991). Computations underlying the execution of movement: a biological perspective. *Science*, 253:287–291.

[Borghi et al., 2016] Borghi, G., Vezzani, R., and Cucchiara, R. (2016). Fast gesture recognition with Multiple Stream Discrete HMMs on 3D skeletons. *Proceedings - International Conference on Pattern Recognition*, pages 997–1002.

[Bouchard and Badler, 2007] Bouchard, D. and Badler, N. (2007). Semantic segmentation of motion capture using laban movement analysis. In *IVA '07 Proceedings of the 7th international conference on Intelligent Virtual Agents*, pages 37–44.

[Caccavale et al., 2019] Caccavale, R., Saveriano, M., Finzi, A., and Lee, D. (2019). Kinesthetic teaching and attentional supervision of structured tasks in human-robot interaction. *Autonomous Robots*, 43(6):1291–1307.

[Carbonera Luvizon et al., 2017] Carbonera Luvizon, D., Tabia, H., and Picard, D. (2017). Learning features combination for human action recognition from skeleton sequences. *Pattern Recognition Letters*, 99:13–20.

[Chiappa and Peters, 2010] Chiappa, S. and Peters, J. (2010). Movement extraction by detecting dynamics switches and repetitions. In *Advances in Neural Information Processing Systems (NIPS)*.

[Daniel et al., 2015] Daniel, C., Kroemer, O., Viering, M., Metz, J., and Peters, J. (2015). Active reward learning with a novel acquisition function. *Autonomous Robots*, 39(3):389–405.

[De Santis et al., 2008] De Santis, A., Siciliano, B., De Luca, A., and Bicchi, A. (2008). An atlas of physical human-robot interaction. *Mechanism and Machine Theory*, 43(3):253–270.

[Deisenroth et al., 2013] Deisenroth, M. P., Neumann, G., and Peters, J. (2013). A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1–2):328–373.

[Diedrichsen et al., 2010] Diedrichsen, J., Shadmehr, R., and Ivry, R. B. (2010). The coordination of movement: optimal feedback control and beyond. *Trends in cognitive sciences*, 14:31–38.

[Dragan et al., 2015] Dragan, A. D., Bauman, S., Forlizzi, J., and Srinivasa, S. S. (2015). Effects of Robot Motion on Human-Robot Collaboration. In *ACM/IEEE International Conference on Human-Robot Interaction*, pages 51–58. IEEE Computer Society.

[Dragan et al., 2013] Dragan, A. D., Lee, K. C., and Srinivasa, S. S. (2013). Legibility and predictability of robot motion. In *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 301–308.

[Fearnhead and Liu, 2007] Fearnhead, P. and Liu, Z. (2007). On-line inference for multiple change point models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69:589–605.

[Flanagan et al., 1999] Flanagan, J. R., Nakano, E., Imamizu, H., Osu, R., Yoshioka, T., and Kawato, M. (1999). Composition and decomposition of internal models in motor learning under altered kinematic and dynamic environments. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 19.

[Flash, 1987] Flash, T. (1987). The control of hand equilibrium trajectories in multi-joint arm movements. *Biol. Cybern*, 57:257–274.

[Flash and Hochner, 2005] Flash, T. and Hochner, B. (2005). Motor primitives in vertebrates and invertebrates. *Current Opinion in Neurobiology*, 15:600–666.

[Flash and Hogan, 1985] Flash, T. and Hogan, N. (1985). The coordination of arm movements: an experimentally confirmed mathematical model. *The journal of Neuroscience*, 5(7):1688–1703.

[Fod et al., 2002] Fod, A., Matrić, M., and Jenkins, O. (2002). Automated derivation of primitives for movement classification. *Autonomous Robots*, 12:39–54.

[Fox et al., 2009] Fox, E., Sudderth, E., Jordan, M., and Willsky, A. (2009). Sharing features among dynamical systems with beta processes. In *Neural Information Processing Systems 22*, pages 549–557. MIT Press.

[Gäbert et al., 2021] Gäbert, C., Kaden, S., and Thomas, U. (2021). Generation of Human-like Arm Motions using Sampling-based Motion Planning. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2534–2541.

[Giszter et al., 1993] Giszter, F., Mussa-Ivaldi, F., and Bizzi, E. (1993). Convergent Force Fields in the Frog's Spinal Cord. *The Journal of Neuroscience*, 13(2):467–491.

[Gomi and Kawato, 1996] Gomi, H. and Kawato, M. (1996). Equilibrium-Point Control Hypothesis Examined by Measured Arm Stiffness During Multijoint Movement. *Science*, 272(5258):117–120.

[Gong et al., 2014] Gong, D., Medioni, G., and Zhao, X. (2014). Structured time series analysis for human action segmentation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36:1414–1427.

[Gong et al., 2012] Gong, D., Medioni, G., Zhu, S., and Zhao, X. (2012). Kernelized temporal cut for online temporal segmentation and recognition. In *European Conference on Computer Vision 2012*, pages 229–243.

[Graybiel, 1998] Graybiel, A. (1998). The basal ganglia and chunking of action repertoires. *Neurobiology of Learning and Memory*, 70:119–136.

[Gulletta et al., 2020] Gulletta, G., Erlhagen, W., and Bicho, E. (2020). Human-like arm motion generation: A review. *Robotics*, 9(4).

[Guo et al., 2014] Guo, X., Singh, S., Lee, H., Lewis, R. L., and Wang, X. (2014). Deep Learning for Real-Time Atari Game Play Using Offline Monte-Carlo Tree Search Planning. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger,

K., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

[Gutzeit, 2021] Gutzeit, L. (2021). A Comparison of Few-shot Classification of Human Movement Trajectories. In *Proceedings of the 10th International Conference on Pattern Recognition Applications and Methods - Volume 1: ICPRAM*, pages 243–250. INSTICC, SciTePress.

[Gutzeit, 2022] Gutzeit, L. (2022). Hierarchical Segmentation of Human Manipulation Movements. In *Proc. of the 26th International Conference on Pattern Recognition*, pages 2742–2748. IEEE.

[Gutzeit et al., 2018a] Gutzeit, L., Fabisch, A., Otto, M., Metzen, J. H., Hansen, J., Kirchner, F., and Kirchner, E. A. (2018a). The BesMan Learning Platform for Automated Robot Skill Learning. *Frontiers in Robotics and AI*, 5.

[Gutzeit et al., 2019] Gutzeit, L., Fabisch, A., Petzoldt, C., Wiese, H., and Kirchner, F. (2019). Automated Robot Skill Learning from Demonstration for Various Robot Systems. In Benzmüller, C. and Stuckenschmidt, H., editors, *KI 2019: Advances in Artificial Intelligence, Conference Proc.*, volume LNAI 11793, pages 168–181. Springer.

[Gutzeit and Kirchner, 2016] Gutzeit, L. and Kirchner, E. A. (2016). Automatic detection and recognition of human movement patterns in manipulation tasks. In *Proceedings of the 3rd International Conference on Physiological Computing Systems*.

[Gutzeit and Kirchner, 2022] Gutzeit, L. and Kirchner, F. (2022). Unsupervised Segmentation of Human Manipulation Movement into Building Blocks. *IEEE Access*, 10:125723–125734.

[Gutzeit et al., 2016] Gutzeit, L., Otto, M., and Kirchner, E. A. (2016). Simple and robust automatic detection and recognition of human movement patterns in tasks of different complexity. In *Physiological Computing Systems*, pages 39–57. Springer.

[Gutzeit et al., 2018b] Gutzeit, L., Tabie, M., and Kirchner, E. A. (2018b). Automatic movement segmentation of exoskeleton data. In *Conference Proceedings of the 3rd International Mobile Brain/Body Imaging Conference, (MoBI-2018)*.

[Hamerly and Elkan, 2004] Hamerly, G. and Elkan, C. (2004). Learning the k in k-means. *Advances in neural information processing*, 16:281–288.

[Hansen and Ostermeier, 2001] Hansen, N. and Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9:159–195.

[Harris and Wolpert, 1998] Harris, C. M. and Wolpert, D. M. (1998). Signal-dependent noise determines motor learning. *Nature*, 394(August):780–784.

[Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

[Hörnstein et al., 2010] Hörnstein, J., Gustavsson, L., Santos-Victor, J., and Lacerda, F. (2010). Multimodal Language Acquisition Based on Motor Learning and Interaction. In Sigaud, O. and Peters, J., editors, *From Motor Learning to Interaction Learning in Robots*, pages 467–489. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Ijspeert et al., 2013] Ijspeert, A. J., Nakanishi, J., Hoffmann, H., Pastor, P., and Schaal, S. (2013). Dynamical movement primitives: learning attractor models for motor behaviors. *Neural Computation*, 25(2):328–373.

[Jenkins and Matarić, 2003] Jenkins, O. C. and Matarić, M.-J. (2003). Automated derivation of behavior vocabularies for autonomous humanoid motion. In *Proceedings of Autonomous Agents and Multi Agent Systems*, pages 225–232.

[Kawato, 1999] Kawato, M. (1999). Internal models for motor control and trajectory planning. *Current Opinion in Neurobiology*, 9(6):718–727.

[Kawato et al., 1987] Kawato, M., Furukawa, K., and Suzuki, R. (1987). A hierarchical neural-network model-for control and learning of voluntary movement. *Biol. Cybern*, 57:169–185.

[Kirchner et al., 2015] Kirchner, E., Fernández, J., Kampmann, P., Schröer, M., Metzen, J., and Kirchner, F. (2015). Intuitive Interaction with Robots-Technical Approaches and Challenges. In Rolf Drechsler, U. K., editor, *Formal Modeling and Verification of Cyber-Physical Systems*, pages 224–248. Springer.

[Kirchner et al., 2018] Kirchner, E. A., Fairclough, S., and Kirchner, F. (2018). Embedded multimodal interfaces in robotics: Applications, future trends and societal implications. In Oviatt, S., Schuller, B., Cohen, P., and Sonntag, D., editors, *Handbook*

of Multimodal-Multisensor Interfaces, volume 3, chapter IX, page n.A. ACM Books, Morgan Claypool, forthcoming.

[Kober et al., 2010] Kober, J., Mülling, K., Krömer, O., Lampert, C. H., Schölkopf, B., and Peters, J. (2010). Movement Templates for Learning of Hitting and Batting. In *IEEE International Conference on Robotics and Automation*.

[Kober and Peters, 2012] Kober, J. and Peters, J. (2012). *Reinforcement Learning in Robotics: A Survey*, volume 12, pages 579–610. Springer.

[Kong and Fu, 2022] Kong, Y. and Fu, Y. (2022). Human Action Recognition and Prediction: A Survey. *International Journal of Computer Vision*, 130:1366–1401.

[Konidaris et al., 2012] Konidaris, G., Kuindersma, S., Barto, A., and Grupen, R. (2012). Robot learning from demonstration by constructing skill trees. *The International Journal of Robotics Research*, 31(3):360–375.

[Koos et al., 2013] Koos, S., Mouret, J.-B., and Doncieux, S. (2013). The Transferability Approach: Crossing the Reality Gap in Evolutionary Robotics. *IEEE Transactions on Evolutionary Computation*, 17(1):122–145.

[Koppenborg et al., 2017] Koppenborg, M., Nickel, P., Naber, B., Lungfiel, A., and Huelke, M. (2017). Effects of movement speed and predictability in human-robot collaboration. *Human Factors and Ergonomics in Manufacturing & Service Industries*, 27(4):197–209.

[Kulić et al., 2012] Kulić, D., Ott, C., Lee, D., Ishikawa, J., and Nakamura, Y. (2012). Incremental learning of full body motion primitives and their sequencing through human motion observation. *The International Journal of Robotics Research*, 31(3):330–345.

[Kumar et al., 2019] Kumar, S., Wöhrle, H., Trampler, M., Simnofske, M., Peters, H., Mallwitz, M., Kirchner, E. A., and Kirchner, F. (2019). Modular Design and Decentralized Control of the Recupera Exoskeleton for Stroke Rehabilitation. *Applied Sciences*, 9(4):626.

[Lacquaniti et al., 1983] Lacquaniti, F., Terzuolo, C., and Viviani, P. (1983). The law relating the kinematic and figural aspects of drawing movements. *Acta psychologica*, 54:115–130.

[Lake et al., 2017] Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. (2017). Building Machines That Learn and Think Like People. *Behavioral and Brain Science*, 40:e253.

[LeCun et al., 2015] LeCun, Y., Hinton, G., and Bengio, Y. (2015). Deep learning. *Nature*, 521:436–444.

[Lichiardopol, 2007] Lichiardopol, S. (2007). *A survey on teleoperation*. DCT rapporten. Technische Universiteit Eindhoven. DCT 2007.155.

[Lin et al., 2016] Lin, J. F. S., Karg, M., and Kulić, D. (2016). Movement Primitive Segmentation for Human Motion Modeling: A Framework for Analysis. *IEEE Transactions on Human-Machine Systems*, 46(3):325–339.

[Lioutikov et al., 2017] Lioutikov, R., Neumann, G., Maeda, G., and Peters, J. (2017). Learning movement primitive libraries through probabilistic segmentation. *The International Journal of Robotics Research*, 36(8):879–894.

[Liu et al., 2017] Liu, J., Shahroudy, A., Xu, D., Kot Chichung, A., and Wang, G. (2017). Skeleton-Based Action Recognition Using Spatio-Temporal LSTM Network with Trust Gates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):3007–3021.

[Mallwitz et al., 2015] Mallwitz, M., Will, N., Teiwes, J., and Kirchner, E. A. (2015). The capio active upper body exoskeleton and its application for teleoperation. In *In Proceedings of the 13th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA-2015)*. ESA.

[Matarić, 2002] Matarić, M. (2002). Sensory-Motor Primitives as a Basis for Imitation: Linking Perception to Action an Biology to Robotics. In Dautenhahn, K. and Nehaniv, C. L., editors, *Imitation in Animals and Artifacts*, chapter 15, pages 391–422. MIT Press, Cambridge, MA, USA.

[Metzen et al., 2013] Metzen, J. H., Fabisch, A., Senger, L., Gea Fernández, J., and Kirchner, E. A. (2013). Towards learning of generic skills for robotic manipulation. *KI - Künstliche Intelligenz*, 28(1):15–20.

[Mitra and Acharya, 2007] Mitra, S. and Acharya, T. (2007). Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37(3):311–324.

[Morasso, 1981] Morasso, P. (1981). Spatial control of arm movements. *Experimental Brain Research*, 42:223–227.

[Morasso and Mussa-Ivaldi, 1982] Morasso, P. and Mussa-Ivaldi, F. (1982). Trajectory formation and handwriting: a computational model. *Biological cybernetics*, 145:131–142.

[Mülling et al., 2013] Mülling, K., Kober, J., Koemer, O., and J.Peters (2013). Learning to select and generalize striking movements in robot table tennis. *The International Journal of Robotics Research*, 32:263–279.

[Munkres, 1957] Munkres, J. (1957). Algorithms for the Assignment and Transportation Problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38.

[Mussa-Ivaldi et al., 1994] Mussa-Ivaldi, F., Giszter, S. F., and Bizzi, E. (1994). Linear combinations of primitives in vertebrate motor control. *Proceedings of the National Academy of Science of the USA*, 91:7534–7538.

[Mussa-Ivaldi and Bizzi, 2000] Mussa-Ivaldi, F. A. and Bizzi, E. (2000). Motor learning through the combination of primitives. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, 355:1755–1769.

[Mussa-Ivaldi and Solla, 2004] Mussa-Ivaldi, F. A. and Solla, S. A. (2004). Neural primitives for motion control. *IEEE Journal of Oceanic Engineering*, 29(3):640–650.

[Nehaniv and Dautenhahn, 2002] Nehaniv, C. L. and Dautenhahn, K. (2002). The Correspondence Problem. In Dautenhahn, K. and Nehaniv, C. L., editors, *Imitation in Animals and Artifacts*, chapter 2, pages 41–61. MIT Press, Cambridge, MA, USA.

[Ng and Russell, 2000] Ng, A. Y. and Russell, S. J. (2000). Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 663–670, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Niekum et al., 2012] Niekum, S., Osentoski, S., Konidaris, G., and Barto, A. (2012). Learning and generalization of complex tasks from unstructured demonstrations. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*.

[Pastor et al., 2009] Pastor, P., Hoffmann, H., Asfour, T., and Schaal, S. (2009). Learning and generalization of motor skills by learning from demonstration. In *2009 IEEE International Conference on Robotics and Automation*, pages 763–768. Ieee.

[Pateria et al., 2021] Pateria, S., Subagdja, B., Tan, A. H., and Quek, C. (2021). Hierarchical Reinforcement Learning: A Comprehensive Survey. *ACM Computing Surveys*, 54(5).

[Patsadu et al., 2012] Patsadu, O., Nukoolkit, C., and Watanapa, B. (2012). Human gesture recognition using Kinect camera. *Computer Science and Software Engineering (JCSSE), 2012 International Joint Conference on*, pages 28–32.

[Peters et al., 2010] Peters, J., Mülling, K., and Altun, Y. (2010). Relative entropy policy search. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*.

[Peters et al., 2012] Peters, J., Mülling, K., Kober, J., Nguyen-Tuong, D., and Krömer, O. (2012). Robot skill learning. In *Proceedings of the European Conference on Artificial Intelligence*.

[Poppe, 2010] Poppe, R. (2010). A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976–990.

[QTM, 2022] QTM (2022). Qualisys track manager, https://www.qualisys.com/software/qualisys-track-manager. [visited on 29/07/2022].

[Ramirez-Amaro et al., 2014] Ramirez-Amaro, K., Beetz, M., and Cheng, G. (2014). Automatic Segmentation and Recognition of Human Activities from Observation based on Semantic Reasoning. In *Intelligent Robots and Systems (IROS)*, pages 5043–5048.

[Ramirez-Amaro et al., 2017] Ramirez-Amaro, K., Beetz, M., and Cheng, G. (2017). Transferring skills to humanoid robots by extracting semantic representations from observations of human activities. *Artificial Intelligence*, 247:95–118.

[Richardson and Flash, 2002] Richardson, M. J. E. and Flash, T. (2002). Comparing smooth arm movements with the two-thirds power law and the related segmented-control hypothesis. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 22:8201–11.

[Rohrer et al., 2004] Rohrer, B., Fasoli, S., Krebs, I., Volpe, B., Frontera, W. R., Stein, J., and Hogan, N. (2004). Submovements grow larger, fewer, and more blended during stroke recovery. *Motor Control*, 8:472–483.

[Sakai et al., 2003] Sakai, K., Kitaguchi, K., and Hikosaka, O. (2003). Chunking during human visuomotor sequence learning. *Experimental Brain Research*, 152(2):229–42.

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

[Schaal, 2007] Schaal, S. (2007). The new robotics–towards human-cententered machines. *HFSP Journal*, 1(2):115–126. PMID: 19404417.

[Schreiter et al., 2015] Schreiter, L., Berghöfer, E., Beyl, T., Senger, L., Raczkowsky, J., Kirchner, F., and Woern, H. (2015). Probabilistic Situation Detection for Human-Robot Interaction in an OP Lab Environment. In *Tagungsband der 14. Jahrestagung der Deutschen Gesellschaft für Computer- und Roboterassistierte Chirurgie e.V, (CURAC-15)*, pages 99–104.

[Schreiter et al., 2014] Schreiter, L., Senger, L., Beyl, T., Berghöfer, E., Raczkowsky, J., and Woern, H. (2014). Probabilistische Echtzeit-Situationserkennung im Operationssaal am Beispiel von OP:Sense. In *Tagungsband der 13. Jahrestagung der Deutschen Gesellschaft für Computer- und Roboterassistierte Chirurgie e.V., (CURAC-2014)*, pages 177–180.

[Senger et al., 2014] Senger, L., Schröer, M., Metzen, J. H., and Kirchner, E. A. (2014). Velocity-based Multiple Change-point Inference for Unsupervised Segmentation of Human Movement Behavior. In *Proceedings of the 22th International Conference on Pattern Recognition (ICPR2014)*, pages 4564–4569.

[Shadmer and Wise, 2005a] Shadmer, R. and Wise, S. P. (2005a). Chapter 8: What

maintains limb stability. In *The Computational Neurobiology of Reaching and Pointing*, chapter 8, pages 119–140. MIT Press.

[Shadmer and Wise, 2005b] Shadmer, R. and Wise, S. P. (2005b). *The Computational Neurobiology of Reaching and Pointing*. MIT Press.

[Shi et al., 2017] Shi, Y., Tian, Y., Wang, Y., and Huang, T. (2017). Sequential Deep Trajectory Descriptor for Action Recognition with Three-Stream CNN. *IEEE Transactions on Multimedia*, 19(7):1510–1520.

[Sosnik et al., 2004] Sosnik, R., Hauptmann, B., Karni, A., and Flash, T. (2004). When practice leads to co-articulation: the evolution of geometrically defined movement primitives. *Experimental Brain Research*, 156(4):422–38.

[Stefanov et al., 2010] Stefanov, N., Peer, A., and Buss, M. (2010). Online intention recognition for computer-assisted teleoperation. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 5334–5339.

[Sternad and Schaal, 1999] Sternad, D. and Schaal, S. (1999). Segmentation of endpoint trajectories does not imply segmented control. *Experimental Brain Research*, 124:118–36.

[Theodorou et al., 2010] Theodorou, E., Buchli, J., and Schaal, S. (2010). A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, 11:3137–3181.

[van der Maaten and Hinton, 2008] van der Maaten, L. and Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

[Viviani and Cenzato, 1985] Viviani, P. and Cenzato, M. (1985). Segmentation and Coupling in Complex Movements. *Jounral of Experimenta Psychology; Human Perception and Performance*, 75:828–845.

[Wächter and Asfour, 2015] Wächter, M. and Asfour, T. (2015). Hierarchical segmentation of manipulation actions based on object relations and motion characteristics. In *Proceedings of the 17th International Conference on Advanced Robotics, ICAR 2015*, volume 270273, pages 549–556.

[Wang et al., 2020] Wang, Y., Yao, Q., Kwok, J. T., and Ni, L. M. (2020). Generalizing from a Few Examples: A Survey on Few-shot Learning. *ACM Computing Surveys*, 53(3).

[Wang et al., 2013] Wang, Z., Mülling, K., Deisenroth, M. P., Amor, H. B., Vogt, D., Schölkopf, B., and Peters, J. (2013). Probabilistic Movement Modeling for Intention Inference in Human-Robot Interaction. *The International Journal of Robotics Research*, 32:841–858.

[Wolpert and Ghahramani, 2000] Wolpert, D. M. and Ghahramani, Z. (2000). Computational principles of motor neuroscience. *Nature Neuroscience*, 3(november):1212–1217.

[Wolpert et al., 1994] Wolpert, D. M., Ghahramani, Z., and Jordan, M. I. (1994). Perceptual distortion contributes to the curvature of human reaching movements. *Experimental Brain Research*, 98(1):153–156.

[Wolpert et al., 1995a] Wolpert, D. M., Ghahramani, Z., and Jordan, M. I. (1995a). Are arm trajectories planned in kinematic or dynamic coordinates? An adaptation study. *Experimental Brain Research*, 103(3):460–470.

[Wolpert et al., 1995b] Wolpert, D. M., Ghahramani, Z., and Jordan, M. I. (1995b). An internal model for sensorimotor integration. *Science*, 269.

[Xsens, 2022] Xsens (2022). MVN Analyze, https://www.xsens.com/products/mvn-analyze. [visited on 29/07/2022].

[Yan et al., 2014] Yan, H., Ang, M. H., and Poo, A. N. (2014). A Survey on Perception Methods for Human-Robot Interaction in Social Robots. *International Journal of Social Robotics*, 6:85–119.

[Zhang et al., 2019] Zhang, H.-B., Zhang, Y.-X., Zhong, B., Lei, Q., Yang, L., Du, J.-X., and Chen, D.-S. (2019). A comprehensive survey of vision-based human action recognition methods. *Sensors*, 19(5).

[Zöllner et al., 2005] Zöllner, R., Pardowitz, M., Knoop, S., and Dillmann, R. (2005). Towards cognitive robots: Building hierarchical task representations of manipulations from human demonstration. *Proceedings - IEEE International Conference on Robotics and Automation*, 2005(April):1535–1540.