Robot Learning from Human Demonstrations for Human-Robot Synergy

Maria Kyrarini

Universität Bremen 2019

Robot Learning from Human Demonstrations for Human-Robot Synergy

Vom Fachbereich für Physik und Elektrotechnik
der Universität Bremen

zur Erlangung des akademischen Grades

Doktor–Ingenieur (Dr.-Ing.)

genehmigte Dissertation

von

Dipl. Ing. / M.Sc. Maria Kyrarini

aus Griechenland

Referent: Prof. Dr.-Ing. Axel Gräser

Korreferent: Prof. Dr.-Ing. Udo Frese

Eingereicht am: 1. April 2019

Tag des Promotionskolloquiums: 28. Mai 2019

Acknowledgements

First of all I would like to express my gratitude to my supervisor Prof. Dr.-Ing. Axel Gräser for offering me a PhD position at Institute of Automation (IAT) and for his valuable guidance, support and wisdom throughout my research work. I would like to thank Prof. Dr.-Ing. Udo Frese for accepting to be the second reviewer of this thesis as well as Prof. Dr.-Ing. Alberto Garcia-Ortiz and Prof. Dr.-Ing. Walter Lang for accepting to be examiners of this thesis.

Further I would like to thank my colleague Dr. Danijela Ristic-Durrant for her support, useful discussions, and constructive feedback regarding my research. Moreover I would like to thank my colleague Mr. Muhammad Abdul Haseeb for all the brainstorming and valuable discussions. I would also like to thank my ex-colleagues Dr. Xingchen Wang, Dr. Xiangpeng Liu, Dr. Adrian Leu, Dr. Sinisa Slavnic, Mr. Felix Goldau, Ms. Qinyuan Fang and Ms. Ge Gao for all the fruitful scientific discussions.

Additionally I would like to thank my colleague Ms. Lena Kredel for sharing her experience regarding assistive robotics with me and for her willingness to participate on some experiments for my thesis. My gratitude goes as well to all the participants to my experiments and to Mr. Michael Ehlen for his technical support. Special thanks go to Mrs. Katrina Stollmann from Fremdsprachenzentrum University of Bremen for proof-reading parts of my thesis and to Mr. Jeroen Schäfer for proof-reading the German abstract. I thank also all students who during the work on their Master Theses and Master Projects contributed to this thesis.

Special thanks go to the Postgraduate International Programme (PIP) in Physics and Electrical Engineering from the University of Bremen for partially funding my participation in conferences and for supporting me to organize two summer schools.

Last by not least, I would like to thank my mother Asimina, my brother Antonis and my grandmother Georgia for their love and support during the past four years. Moreover, I would like to express my gratitude and love to my husband Rehan for his unconditional support and for proof-reading my thesis. My deepest gratitude goes to all of my friends in Bremen, in Athens and around the world.

Abstract

Human-robot synergy enables new developments in industrial and assistive robotics research. In recent years, collaborative robots can work together with humans to perform a task, while sharing the same workplace. However, the teachability of robots is a crucial factor, in order to establish the role of robots as human teammates. Robots require certain abilities, such as easily learning diversified tasks and adapting to unpredicted events. The most feasible method, which currently utilizes human teammate to teach robots how to perform a task, is the Robot Learning from Demonstrations (RLfD). The goal of this method is to allow non-expert users to 'program' a robot by simply guiding the robot through a task.

The focus of this thesis is on the development of a novel framework for Robot Learning from Demonstrations that enhances the robots' abilities to learn and perform the sequences of actions for object manipulation tasks (high-level learning) and, simultaneously, learn and adapt the necessary trajectories for object manipulation (low-level learning). A method that automatically segments demonstrated tasks into sequences of actions is developed in this thesis. Subsequently, the generated sequences of actions are employed by a Reinforcement Learning (RL) from human demonstration approach to enable high-level robot learning. The low-level robot learning consists of a novel method that selects similar demonstrations (in case of multiple demonstrations of a task) and the Gaussian Mixture Model (GMM) method. The developed robot learning framework allows learning from single and multiple demonstrations.

As soon as the robot has the knowledge of a demonstrated task, it can perform the task in cooperation with the human. However, the need for adaptation of the learned knowledge may arise during the human-robot synergy. Firstly, Interactive Reinforcement Learning (IRL) is employed as a decision support method to predict the sequence of actions in real-time, to keep the human in the loop and to enable learning the user's preferences. Subsequently, a novel method that modifies the learned Gaussian Mixture Model (m-GMM) is developed in this thesis. This method allows the robot to cope with changes in the environment, such as objects placed in a different from the demonstrated pose or obstacles, which may be introduced by the human teammate. The modified Gaussian Mixture Model is further used by the Gaussian Mixture Regression (GMR) to generate a trajectory, which can efficiently control the robot.

The developed framework for Robot Learning from Demonstrations was evaluated in two different robotic platforms: a dual-arm industrial robot and an assistive robotic manipulator. For both robotic platforms, small studies were performed for industrial and assistive manipulation tasks, respectively. Several Human-Robot Interaction (HRI) methods, such as kinesthetic teaching, gamepad or 'hands-free' via head gestures, were used to provide the robot demonstrations. The 'hands-free' HRI enables individuals with severe motor impairments to provide a demonstration of an assistive task. The experimental results demonstrate the potential of the developed robot learning framework to enable continuous human–robot synergy in industrial and assistive applications.

Kurzfassung

Die Mensch-Roboter-Synergie ermöglicht neue Entwicklungen in der industriellen und assistiven Robotikforschung. In den letzten Jahren können kollaborative Roboter mit Menschen zusammenarbeiten, um eine Aufgabe zu erfüllen, während sie sich den gleichen Arbeitsplatz teilen. Allerdings ist die Lernfähigkeit von Robotern ein entscheidender Faktor, um die Rolle der Roboter als Kollegen des Menschen zu etablieren. Die Roboter benötigen die Fähigkeit, verschiedenartige Aufgaben leicht zu erlernen und sich an unvorhergesehene Ereignisse anzupassen. Die derzeit praktikabelste Methode ist das Robot Learning from Demonstrations (RLfD). Diese setzt menschliche Kollegen ein, um Robotern beizubringen, wie man eine Aufgabe erfüllt. Das Ziel dieser Methode ist es, Nicht-Experten zu ermöglichen, einen Roboter zu 'programmieren', indem sie den Roboter durch eine Aufgabe führen.

Der Schwerpunkt dieser Arbeit liegt auf der Entwicklung eines neuartigen Frameworks für RLfD, das die Roboterfähigkeiten erweitert, um die Sequenzen von Aktionen für Objektmanipulationsaufgaben zu lernen und durchzuführen (High-Level-Learning) und gleichzeitig die notwendigen Trajektorien für Objektmanipulationen zu erlernen und anzupassen (Low-Level-Learning). In dieser Arbeit wird ein Verfahren entwickelt, das demonstrierte Aufgaben automatisch in eine Abfolge von Aktionen unterteilt. Anschließend werden die erzeugten Handlungsabläufe von einem Reinforcement-Lernen-Algorithmus (RL) aus dem menschlichen Demonstrationsansatz genutzt, um ein hochrangiges Robot Learning zu ermöglichen. Das Low-Level-Robot-Learning besteht aus einem neuartigen Verfahren, das ähnliche Demonstrationen auswählt (im Falle mehrerer Demonstrationen einer Aufgabe) und Gaussian Mixture Model (GMM) Verfahren. Das entwickelte Robot-Learning-Framework ermöglicht das Lernen aus einzelnen und mehreren Demonstrationen.

Sobald der Roboter das Wissen über eine demonstrierte Aufgabe hat, kann er diese in Zusammenarbeit mit dem Menschen ausführen. Die Notwendigkeit einer Anpassung des erlernten Wissens kann jedoch während der Mensch-Roboter-Synergie entstehen. Zuerst wird Interaktives Reinforcement Lernen (IRL) als entscheidungsunterstützende Methode eingesetzt, um den Ablauf von Aktionen in Echtzeit vorherzusagen, den Menschen auf dem Laufenden zu halten und das Erlernen der Präferenzen des Benutzers zu ermöglichen. Anschließend wird in dieser Arbeit eine neuartige Methode entwickelt, die das erlernte GMM modifiziert (m-GMM). Dieses Verfahren ermöglicht es dem Roboter, Veränderungen in der Umgebung zu bewältigen, wie z. B. Objekte, die sich in einer anderen als der gezeigten Pose befinden oder Hindernisse, die

vom menschlichen Kollegen eingeführt werden können. Das modifizierte GMM wird von der Gaussian Mixture Regression (GMR) weiterverwendet, um eine Trajektorie zu erzeugen, mit welcher der Roboter effizient gesteuert werden kann.

Das entwickelte Framework für RLfD wurde mit zwei verschiedenen Roboterplattformen evaluiert: einem zweiarmigen Industrieroboter und einem assistiven Roboter-Manipulator. Für beide Roboterplattformen wurden kleine Studien für industrielle bzw. assistive Manipulationsaufgaben durchgeführt. Mehrere Mensch-Roboter-Interaktions-Methoden (MRI), wie kinästhetischer Unterricht, Gamepad oder 'hands-free', durch Kopfbewegungen, wurden eingesetzt, um die Roboterdemonstration durchzuführen. Die ‚hands-free' MRI-Methoden ermöglicht es Personen mit schweren motorischen Beeinträchtigungen eine assistive Aufgabe zu demonstrieren. Die experimentellen Ergebnisse zeigen ein Potenzial des entwickelten Robot-Learning-Frameworks, um eine kontinuierliche Mensch-Roboter-Synergie in industriellen und assistiven Anwendungen zu ermöglichen.

Table of Contents

Acknowledgements iii
Abstract v
Kurzfassung vii
1. Introduction 1
1.1 Background 1
1.2 Problem Statement 2
1.3 Contributions 3
1.4 Thesis Overview 5
2. Machine Learning Techniques for Robot Learning from Human Demonstrations 7
2.1 Machine Learning Techniques 7
2.1.1 Supervised Learning 8
2.1.2 Unsupervised Learning 9
2.1.3 Reinforcement Learning (RL) 10
2.2 Robot Learning from Human Demonstrations 11
2.2.1 High-Level Symbolic Task Learning 13
2.2.2 Low-Level Skill Learning at Trajectory Level 16
2.2.3 High-Level Task and Low-Level Skill Robot Learning 20
2.3 Robot Learning by Individuals with severe Motor Impairments 22
3. Robot Learning of Object Manipulation Tasks from Human Demonstrations 23
3.1 Overview of the Robot Learning Framework 23
3.2 Human-Robot Interaction for Robot Learning 26
3.2.1 'Hand-Operated' Human-Robot Interaction for Robot Learning 26
3.2.2 'Hands-free' Human-Robot Interaction for Robot Learning 27
3.3 Robot Learning Phase (Offline) 33
3.3.1 Symbolic Task Learning (High-level) 34
3.3.2 Skill Learning at trajectory level (Low-level) 37
3.4 Robot Working Phase (Online) 45
3.4.1 Decision Support (High-level) 46
3.4.2 Adaptation at Trajectory Level (Low-level) 49
3.4.3 Robot Learning User's Preferences 57
4. Robot Learning of Industrial Assembly Tasks from Multiple Human Demonstrations – Application in Industrial Robotics 59
4.1 Industrial Robotic Platform 59
4.2 Robot Gripper Assembly Task (Task 1) 60
4.2.1 Human Demonstrations of the Robot Gripper Assembly Task 61
4.2.2 Robot Learning and Working Phase of Robot Gripper Assembly Task 62

4.3 'Pins into Holes' Task (Task 2) 80
4.3.1 Human Demonstrations of 'Pins into Holes' Task 80
4.3.2 Robot Learning and Working Phase of the 'Pins into Holes' Task 83
4.4 Discussion 92
5. Robot Learning of an Assistive Manipulation Task from One-shot Human Demonstration – Application in Assistive Robotics 95
5.1 Assistive Robotic Platform 95
5.2 Assistive Manipulation Task for Serving a Drink 96
5.3 Human Demonstration of the Assistive Manipulation Task 97
5.4 Robot Learning and Working Phase of the Assistive Manipulation Task 99
5.5 Discussion 105
6. Conclusions and Outlook 107
6.1 Thesis Summary and Conclusions 107
6.2 Outlook 109
Abbreviations 111
Bibliography 112
Appendix A: Publications by the author 125
Appendix B: Polyline Simplification Ramer-Douglas-Peucker (RDP) 128
Appendix C: Experimental Results in Industrial Robotics Application 129
I. Robot Gripper Assembly Task (Task 1) 129
II. 'Pins into Holes' Task (Task 2) 153
Appendix D: Experimental Results in Assistive Robotics Application 163

1. Introduction

1.1 Background

In the last decade, there has been a growing interest in humans and robots working together in several fields, like manufacturing [8], search-and-rescue [9] [10], surgical robotics [11], and service and assistive robotics [12], [13]. In the past, robotic research was focused on creating machines able to perform repetitious tasks with high precision and speed, skills which are required by the industrial automation domain. Robotic manipulators were kept in isolation behind 'cages' and they were pre-programmed to perform a specific task in a well-structured and constant environment. The focus of robotic research today has shifted toward human-centered applications, in which the robots are not seen as independent 'workers' but rather as cooperative human partners.

Collaborative robotic manipulators, so called Cobots, are starting to be part of the industrial world [14]. The main idea of human-robot synergetic work is to combine the advantages of both teammates. While the Cobot handles heavy lifting and performs repetitive actions with high precision, its human teammate simultaneously takes care of tasks that demand dexterity, flexibility and cognition. Some factories [15], [16], [17], [18] in Germany have already deployed Cobots to work alongside humans.

An important aspect of Cobots is their teachability. In a traditional programming scenario, a human with expertise in programming would have to take into account all the possible events in advance, and would need to code the actions of the robot as a response to all the different events [1]. The robot has to be tested for robustness for all possible events. If a failure occurs or if new circumstances appear, high-skilled programmers have

to update the robot code. However, during human-robot synergetic work, programming of all the possible events is not viable. The human brings the element of uncertainty into the shared workspace, which is hard to predict and properly code in the robot program. For example, the human co-worker may position or orient some objects differently or may place some additional objects by mistake.

In the last few decades, an alternative approach in robotic research has been Robot Learning from Demonstration [19]. The main principle is that end-users can teach the robots new tasks by demonstration, as opposed to time-consuming and highly technical skills demanding traditional robot programming methods [20]. Easily teachable robots learn from demonstration, which does not require any expert knowledge of robotics technology by the end-user, and could benefit industrial, social, service and assistive robots.

1.2 Problem Statement

The state-of-the-art in robot learning from human demonstrations is presented in various survey papers [21] [22] [23] [24] [25] [26]. Robot Learning from demonstration (RLfD) is not a record and play technique, but implies learning, and generalization [22]. Three processes of robot learning are defined by Bakker & Kuniyoshi in [27] and by Zhu & Hu in [24] as:

- observe an action (sensing),
- represent the action (understanding),
- and reproduce the action (doing).

With the rapid advancement in machine learning techniques, RLfD started incorporating more of those techniques to represent (i.e. how to generalize across demonstrations) and reproduce the demonstrated action (i.e. to generalize the movement to new situations) [22]. The state-of-the-art in RLfD is comprehensively discussed in chapter 2.

Despite the progress in machine learning and RLfD, there are many open issues. Some of the ongoing challenges are summarized as follows:

1. *Bridging the gap between high-level symbolic reasoning and low-level motor control learning:* Current research in RLfD tends to focus either on continuous representations of the low-level motor control of robots (*low-level skill learning at trajectory level* [1]) or on discrete representations of the high-level task requirements (*high-level symbolic task learning*[2]) [23] [25] [26].
2. *Coping with conflicting demonstrations across teachers:* Several human teachers may demonstrate the task with different styles. Humans may perform the same task with different preferences. There is limited work in RLfD in this domain as the demonstrations are provided with an explicit concept of the task [22].
3. *Exploiting the social interaction in RLfD:* Humans have the ability to evaluate the robot performance. RLfD has to cope with the questions of how and when a robot should request user feedback and how to use this information [23] [21].

1.3 Contributions

This thesis aims at contributing to the field of Robot Learning from Demonstrations (RLfD) by proposing a holistic approach to the RLfD for object manipulation, while presenting solutions for the open issues mentioned above.

Challenge 1: Bridging the gap between high-level symbolic reasoning and low-level motor control learning

The developed RLfD framework learns the sequence of actions (high-level) needed to perform the demonstrated task, including the trajectories (low-level). During human and robot synergetic work, the framework adapts the learned trajectories in real-time to environmental changes (new object positions and obstacles), which are introduced by the user, without additional training. To illustrate:

[1] Low-level skill learning at trajectory level: this term represents robot learning of motor control (motion primitives or trajectories).

[2] High-level symbolic task learning: this term represents robot learning of sequence of actions needed to perform a task (symbolic reasoning)

- The developed RLfD framework identifies object pick&place actions and associates the actions to the manipulated object (High-level learning). – *Section 3.3.1.1*
- The Gaussian Mixture Model (GMM) algorithm is used for learning the constraints of the demonstrated trajectories (Low-level learning). – *Section 3.3.2.2*
- The novel algorithm m-GMM (modification of GMM) is developed in this thesis to modify the learned GMM to the environmental changes. – *Section 3.4.2*

Challenge 2: Coping with conflicting demonstrations across teachers

This thesis proposes two approaches to cope with conflicting demonstrations in high-level and low-level learning. To Illustrate:

- Reinforcement learning is used to learn the different sequences of actions (high-level) demonstrated by teachers. – *Sections 3.3.1.2 and 3.4.1*
- A novel algorithm is developed which selects similar demonstrated trajectories and deals with the variety in speed between different demonstrations (low-level). – *Section 3.3.2.1*

Challenge 3: Exploiting the social interaction in RLfD

To exploit social interaction, the developed RLfD framework uses interactive reinforcement learning to keep the human in the loop. The robot learns user's preferences regarding the sequence of actions (high-level) by requesting user feedback. – *Section 3.4.3*

Other contributions of the RLfD framework developed in this thesis include:

- Several Human-Robot Interaction (HRI) methods are used to provide the robot demonstrations. The HRIs are performed by hands or by head motion ('hands-free') and are used to directly control the robot's end-effector. 'Hands-free' HRI can enable individuals with severe motor impairments, such as tetraplegia[3], to provide a demonstration of a desired task. – *Section 3.2*

[3] Tetraplegia (also known as quadriplegia) is the paralysis of all 4 extremities (arms and legs) due to an injury or illness.

- The RLfD framework is able to learn by single or multiple demonstrations depending on the user's capabilities and the purpose of the manipulation task – assistive or industrial. – *Section 3.1*
- The RLfD framework enables the robot to perform a new task using the gained knowledge from previous tasks. – *Section 3.4.1*
- The RLfD framework is generic. For proof-of-concept, the framework is implemented in two different robotic platforms: a dual-arm industrial robot and an assistive robotic manipulator. For both robotic platforms, real-world experiments were performed for industrial and assistive manipulation tasks, respectively. – *Chapters 4&5*

1.4 Thesis Overview

This thesis is structured in the following chapters:

- Chapter 2 presents the concepts of RLfD. Firstly, a brief introduction to machine learning techniques is given, after which the state-of-the-art algorithms in RLfD and its applications are described.
- Chapter 3 presents the developed RLfD framework. Firstly, it introduces the HRI methods which are used in this thesis. Subsequently, it explains the offline learning phase and the online working phase of the RLfD framework.
- Chapter 4 shows the real-world experiments of the RLfD framework in industrial applications. The robotic platform is a dual-arm industrial robot and two industrial assembly tasks are selected to validate the framework.
- Chapter 5 presents the real-world experiments of the RLfD framework in an assistive application. The robotic platform is an assistive robot and a tetraplegic user provides a demonstration of a manipulation task using a 'hands-free' HRI. The results of these experiments are discussed in this chapter.
- Chapter 6 summarizes the conclusions of this thesis and discusses new possible routes of research arising from the presented work.

2. Machine Learning Techniques for Robot Learning from Human Demonstrations

In this chapter, an introduction to machine learning is given. Supervised and unsupervised machine learning techniques are presented as well as reinforcement learning. Moreover, the state-of-the-art in robot learning from human demonstrations is discussed in detail.

2.1 Machine Learning Techniques

The field of Machine Learning (ML) addresses the question of how a machine can automatically learn and improve with experience. Arthur Samuel, a pioneer in Machine Learning, described it as the "field of study that gives computers the ability to learn without being explicitly programmed" [28]. In 1997, Mitchel in [29] defined Machine Learning as follows: "*A computer program is said to* ***learn*** *from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks on T, as measured by P, improves with experience E*". A Machine Learning algorithm uses as input training data, representing the experience of some tasks, and its output is some expertise in the tasks.

Machine Learning techniques have various applications, including web search, email anti-spam, speech recognition, product recommendations, computer vision, robotics, and more [30]. The process of developing Machine Learning algorithms may consist of the

following steps [31]: data collection, data preprocessing, feature extraction from the data, ML training, and ML testing (validation). Machine learning can be classified into the following three categories [32]: supervised, unsupervised and reinforcement learning.

2.1.1 Supervised Learning

Supervised Learning [33] intends to learn a mapping function from examples of input-output pairs. Supervised Learning algorithms use a labeled training dataset to predict a model (mapping function), which represents the patterns that exist in the training dataset. The goal of Supervised Learning is to approximate a model that the output data (labels) can accurately be predicted from new input data. The performance of the Supervised Learning algorithm is evaluated with testing data that do not belong in the training dataset.

Supervised Learning algorithms can be grouped into regression and classification. In regression, the labels are continuous values (real values), while in classification the labels are a finite set of values (classes). Figure 1a illustrates an example of regression, where the algorithm uses as input the pairs of input-output values and it predicts a model that fits the data. For new inputs, the algorithm will estimate output values based on the learned model. Figure 1b illustrates an example of classification, where labeled data of two classes are given as input. The algorithm finds a model, which separates the two classes. For new inputs, the algorithm will predict in which class they belong based on the learned model.

Linear, polynomial, logistic, and nonlinear regression, random forests, and Neural Networks are some machine learning techniques that solve regression problem. Support Vector Machine (SVM), K-Nearest Neighbors (KNN), random forests, Neural Networks, Hidden Markov Models, (HMM) and decision trees, are some popular classifiers.

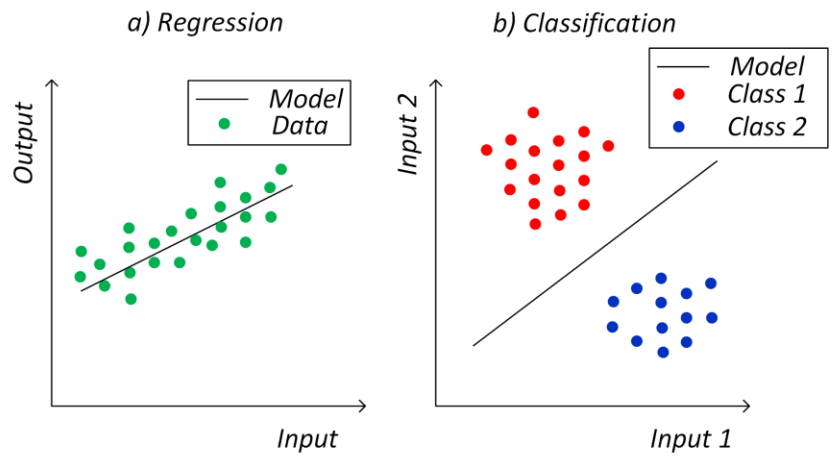


Figure 1: Explanatory illustration a) Regression, b) Classification.

2.1.2 Unsupervised Learning

In Unsupervised Learning [32], the input data are unlabeled and the goal of the learning algorithm is to group the input data based on their similarity (clustering). Figure 2 illustrates a clustering example. Some popular Unsupervised Learning algorithms are mixture models, Kss-means, Neural Networks, Hidden Markov Model (HMM), Density-based spatial clustering, and hierarchical clustering. As it can be observed, some algorithms can be used in a supervised and unsupervised manner, depending on the problem that they try to solve.

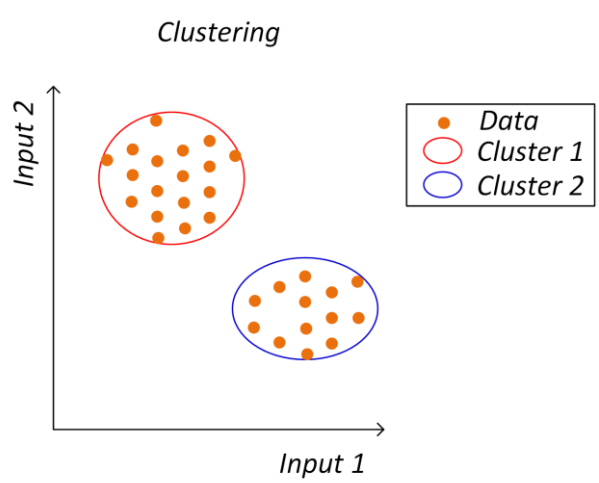


Figure 2: Explanatory illustration of Clustering.

2.1.3 Reinforcement Learning (RL)

Reinforcement Learning (RL) is defined by Sutton & Barto in [34] as follows: "*Reinforcement Learning is learning what to do - how to map situations to actions - so as to maximize a numerical reward signal. The learner is not told which actions to take, but instead must discover which actions yield the most reward by trying them.*" In RL an agent (learner) discovers an optimal behavior (policy) by trial-and-error interactions (reward/punishment) with its environment. Reinforcement Learning in robotics suffers from the 'Curse of Dimensionality' [35], due to high dimensional states and actions of robots.

The Reinforcement Learning consists of the following elements: a set of states that can describe the environment, a set of fixed number of actions that the agent can take, and a reward value that is given to the agent by the environment. At each state, the agent observes the environment and selects an action to take. After taking the action, the environment provides a reward to the agent. The interaction between the agent and the environment is illustrated in Figure 3.

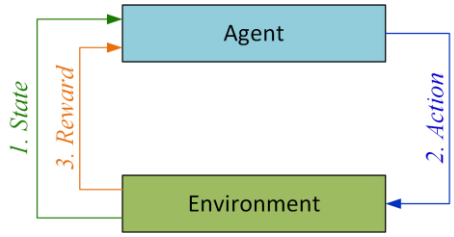


Figure 3: The agent-environment interaction in Reinforcement Learning. (adapted from [34])

The goal of Reinforcement Learning is to use the State – Action – Reward – State and learn a mapping from states to a measured long-term value, known as the optimal value function [36]. Several methods, such as Q-learning, SARSA, and Monte Carlo, have been developed to estimate the optimal value function [34].

2.2 Robot Learning from Human Demonstrations

Robot learning from demonstrations (RLfD) explores techniques that enable robots to learn new skills by a human teacher (demonstrator). RLfD is also called Programming by Demonstration or Imitation Learning [19]. At the beginning of the 1980s, RLfD started attracting attention in manufacturing robotics, as it appeared to be a promising alternative to automate the tedious programming (coding) of the robots [37] [38]. The RLfD progressively moved from simply repeating the demonstrated movements to generalizing across demonstrations using machine learning techniques [19]. Several techniques proposed in machine learning could be applied to robotic data (sensory, actuators) to provide a generalized model. The applications of machine learning tools that are applied in RLfD are discussed in this chapter.

The RLfD is composed of three fundamental steps: *observation*, *representation* and *reproduction* [27]. There are different methods to *observe a demonstration* and gather the datasets as summarized By Argall et al. in [21].

- *Teleoperation*: The human teacher operates the robot and the sensors of the robot record the execution. The recorded data by the sensors of the robot are later used for robot learning. One method of teleoperation is by using a joystick. The human uses the joystick to guide the robot through the task, while the sensors of the robot record the demonstration. Another method of providing the demonstration is by *kinesthetic teaching* [39] [40], where the robot is physically guided through the task by the human and the robot movements are recorded by the robot sensors. This method works for lightweight robots or robots driven by gravity-compensation controllers. Another approach to operate the robot for a manipulation task is by using natural language commands [41]. The human teacher instructs the robot with spatial language, using object reference and positional and directional prepositions. ‘Hands-free’ human-robot interfaces based on eye gaze [42] [43], brain signals using Brain-Computer Interface (BCI) [44] [45], and head motion measured by an Inertial Measurement Unit (IMU) [46] can be potentially used to provide robot demonstrations.

- *Sensors placed on the teacher's body*: Sensors placed on the teacher's body are used to record the demonstration of a task. The teacher is equipped with wearable motion sensors, like IMU [47] or electromyographic (EMG) sensors [48] or cyber-glove [49] [50] or combination of accelerometers and EMG sensors [51]. The teacher's movements are recorded from the wearable sensors and the recorded data can be later used for robot learning.
- *External observations*: External sensors that are not placed on the teacher and the robot are used to record the demonstration of a task. Typically, vision-based external sensors [52] [53] are used to record the teacher's movements during the demonstration.

Given a dataset of a demonstration which have been acquired using one or combination of the above mentioned methods, the robot is able to learn the demonstrated task from the dataset. In some studies, instead of using one dataset, datasets of multiple demonstrations are used [54] [55].

There are different approaches to represent and reproduce the demonstrated task. These approaches are grouped by Argall et al. in [21] and by Billard et al. in [19] into two categories: *low-level skill learning at trajectory level* (or low-level motor control learning) and *high-level symbolic task learning* (or high-level symbolic reasoning). Table 1 summarized the advantages and drawbacks of the two categories.

Table 1: Advantages and drawbacks of high-level and low-level robot learning. (adapted from [19])

Robot Learning	Generalization Process	Advantages	Drawbacks
High-level symbolic task learning	Sequential presentation of pre-defined actions	Allows to learn hierarchy and rules	Requires a large amount of predefined actions for the task segmentation and reproduction
Low-level skill learning at trajectory level	Generic representation of motion (movements)	Allows encoding of very different types of signals/gestures	No reproduction of complicated high-level tasks

2.2.1 High-Level Symbolic Task Learning

In high-level symbolic task learning, the task is encoded according to the sequences of predefined actions. This approach allows the robot to learn the sequence of actions, so the robot can learn high-level tasks [19]. It relies on a priori knowledge to be able to abstract the important key-points of the demonstrated task [1].

Typically, an approach for symbolic task learning is to assume a pre-defined mapping of sensor data to the objects involved in the task. A *graph-based approach* was presented by Nicolescu & Mataric in [56], which generalizes the transportation of an object by a mobile robot. The tasks were generalized in a hierarchical manner. Each node of the graph represented a predefined action (behavior).

A similar *graph-based approach* was presented by Ekvall & Kragic in [55]. The robot learned the high-level constraints of manipulation tasks by multiple demonstrations in three steps. The proposed solution first segmented the demonstrated task into primitive tasks, which was a composite of predefined actions. Subsequently, the state generation modeled the subtasks as states and searched for similar states over multiple demonstrations. At last, the task generalization identified the task constraints and the order of the states for reaching the task goal. The approach was used to encode household tasks, such as setting up the table by a robot manipulator. Similarly in [57], Pardowitz et al. proposed task precedence graphs, which represented the sequential structure of a demonstrated task.

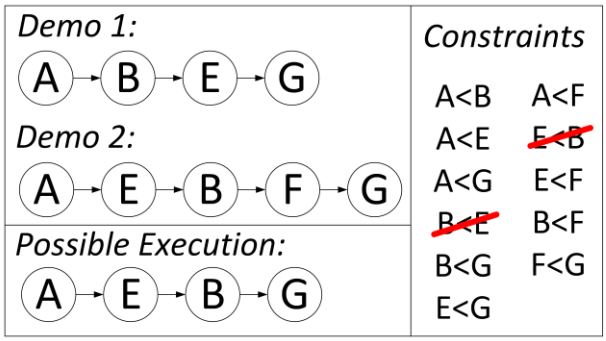


Figure 4: Graph-based approach of representing high-level sequence of actions for a demonstrated task. Each letter represents an object manipulation. Top left: Two demonstrations given to the robot. Right: Extraction of the task constraints. Note that the constraints B<E and E<B are removed. Bottom left: One of possible sequences to follow at execution time. (adapted from [55])

Besides the graph-based approaches, *First-Order Predicate Logic (FOPL)* provides a high-level representation of a task. A sentence in FOPL is presented in the form P(x), where P is the predicate (i.e. robot action) and x is the subject (i.e. manipulated objects). Complete sentences are logically built following the same rules as those used in Boolean algebra. In [58], Cubek et al. used FOPL to describe the goals of a manipulation task. The robot learning method extracted the key states from demonstrations and recognized the goals, which were translated to FOPL. As FOPL implementation, the Planning Domain Definition Language (PDDL) [59] was used. The practical applicability of the method was shown in a setting up table scenario. In the same fashion, PDDL was used by Qian et al. in [60] to model the semantics of human demonstrations for two tasks – pouring water and installation cartridge.

The concept of context-free grammars has been proposed for modeling structured processes [61] [62]. Yang et al. in [63] proposed a computational linguistics framework that learns semantics of manipulation tasks. The framework was based on a *Combinatory Categorial Grammar (CCG)*. A large available dataset of manipulation tasks was used to validate the approach.

Machine learning approaches have been also used to represent high-level tasks in abstracted form. In [64], Hovland et al. adopted *Hidden Markov Models (HMMs)* to provide symbolic representations of assembly tasks. A probabilistic approach for the representation and learning of complex manipulation based on multi-level *Hierarchical Hidden Markov Model (HHMM)* was presented by Patel et al. in [65], where the complex manipulation tasks were decomposed into multiple levels of abstraction to represent the actions in simpler way called action primitives.

In [66], Lu et al. proposed a framework that combines *Knowledge Representation and Reasoning (KRR)* with model-based *Reinforcement Learning (RL).* The robot was able to reason with declarative knowledge and to simultaneously learn from interaction with the environment. To validate the framework, experiments were conducted using a mobile robot on delivery tasks.

In this thesis, a similar approach to [66] is followed. The robot learning framework automatically generates the sequence of actions of the demonstrated task from the human

demonstrations. Subsequently, *Interactive Reinforcement Learning (IRL)* is used to predict the sequence of actions in real-time, to keep the human in the loop and to enable learning the user's preferences. The RL provides an ideal framework for sequential decision making. In the IRL, the teacher can interact with the robot and can influence or evaluate the robot's decision making [67]. The combination of RLfD and RL reduces the exploration problem of classical RL approaches. Table 2 summarizes the above mentioned related work.

Table 2: Overview of Related Work in High-Level Symbolic Task Learning.

Method	**Evaluation task**	**Reference**
Graph-based approach	transportation of an object	Nicolescu & Mataric, 2003 [56]
Graph-based approach	setting up the table	Ekvall & Kragic, 2006 [55]
Graph-based approach	setting up the table	Pardowitz et al., 2007 [57]
First-Order Predicate Logic (FOPL)	setting up the table	Cubek et al., 2015 [58]
Planning Domain Definition Language (PDDL)	pouring water and installation cartridge	Qian et al., 2018 [60]
Combinatory Categorial Grammar (CCG)	object manipulation	Yang et al., 2015 [63]
Hidden Markov Model (HMM)	assembly	Hovland et al., 1996 [64]
Hierarchical Hidden Markov Model (HHMM)	pouring water, handover, use of hammer, spraying from a spray bottle, drinking from a mug, shift object, sprinkle salt	Patel et al., 2014 [65]
Knowledge Representation and Reasoning - Reinforcement Learning (KRR-RL)	delivery tasks	Lu et al., 2018 [66]
Interactive Reinforcement Learning	assembly, pins-into-holes, drinking from a glass with a straw	Presented in this thesis (Sections 3.3.1, 3.4.1 & 3.4.3)

2.2.2 Low-Level Skill Learning at Trajectory Level

The main goal of low-level skill learning at trajectory level (also called motor control learning) is to enable robot to learn the basic movements or gestures (motor skills). However, this approach does not allow reproducing of more complicated high-level tasks. An extensive survey on learning of robot motion has been published by Calinon & Lee in [25]. The most well-known methods are discussed in this section.

A line of research has adapted a dynamical system approach to deal with the demonstrated trajectory, as well as to reproduce it with a new goal pose. One popular method is the *Dynamic Movement Primitive (DMP)* [68] [69] [70], which allows the robot to learn a non-linear differential equation based on the movement observed by a single demo. The equations form a control policy to generate a trajectory for a new goal pose. *Modified DMP* methods proposed by Park et al. in [71] and by Pastor et al. in [72], adapt the learned differential equation for different start and goal positions of a movement or for obstacle avoidance. Another approach to achieve generalization is the *mixture of motor primitives (MoMP)* algorithm proposed by Muelling et al. in [73]. The presented method generates a set of DMPs from demonstrations (each demonstration is represented by one DMP) and then generalizes these movements to a wide range of situations. To evaluate a method, striking movements in robot table tennis were selected.

Furthermore, DMPs were combined with *Reinforcement Learning (RL)* for rhythmic primitives by Kober & Peters in [40]. The demonstration was first described by DMP. Subsequently, policy learning by weighting exploration with the returns (best-suited RL) was used by the robot to reproduce two complex motor tasks, i.e. Ball-in-a-cup and Ball-Paddling. A similar approach was presented by Kormushev et al. in [74], where Expectation-Maximization based RL was exploited to modulate the DMPs of a pancake-flipping movement initialized from human demonstration. In the same fashion, Deng et al. in [75] applied DMPs and RL on a dual-arm mobile manipulator. The RL was employed to adjust the learned DMP model to uncertain environments.

The DMP algorithm models the motion primitive by a single demonstration. To enable learning from multiple demonstrations, Yin & Chen in [76] and Chen et al. in [77]

combined the DMP with *Gaussian Mixture Model (GMM)*. The GMM is used to learn the joint probability of the non-linear system and *Gaussian Mixture Regression (GMR)* is used to reproduce the trajectory. The proposed method was tested on a non-holonomic mobile robot. Another approach to model the robot motion primitives as dynamic systems is *Stable Estimator of Dynamic Systems (SEDS)* and it was proposed by Khansari-Zadeh & Billard in [78]. SEDS is an optimization approach for statistically encoding a dynamic motion as a first order autonomous non nonlinear differential equation with GMM. The method ensures global asymptotic stability of the autonomous nonlinear dynamic system.

Machine learning techniques have been widely used for learning at trajectory level. *Hidden Markov model (HMM)* is one of the methods, which learns the time and space constraints of a given trajectory. Brand & Hertzmann in [79] used HMM to encode and synthesize common elements in a motion. In the same fashion, Calinon et al. in [80] and Billard et al. in [81] modeled the trajectories into HMM by decomposing them into a set of relevant key points and generated the trajectory using spline fitting and interpolation, respectively. In [54] and [82], Calinon el al. proposed the use of GMR to obtain a smooth generalized trajectory. The robot reproduction satisfied the constraints of the demonstrated trajectories. In [83], Pignat & Calinon presented *Hidden Semi-Markov Model (HSMM)* coupled with a *task-parameterized model (TP)* to achieve adaptation to different environmental situations, such as assistive dressing of people with disabilities. The method requires demonstrations with different environmental conditions, so to achieve a generalized model of the motion.

In [84], Calinon el al. proposed the use of *Gaussian Mixture Model (GMM)* to learn the time and space constraints of multiple demonstrations and *Gaussian Mixture Regression (GMR)* is used to reproduce the trajectory. It was shown that GMM represents the constraints along the trajectories continuously, while HMM represents the relevant key points of the trajectories. The advantage of the GMM/GMR over HMM/GMR is that generated trajectories are smoother. In [85] and [86], Calinon proposed the *Task-Parameterized GMM (TP-GMM)* to automatically adapt the movements to new situations. The method requires demonstrations with different environmental conditions, so to

achieve a generalized model of the motion. However, how many demonstrations are needed for different environmental conditions is unclear. TP-GMM has been applied to physical human-robot collaborative movements [87].

In [88] and [89], Vecerik et al. and Hester et al., combined *Deep Reinforcement Learning (DRL)* with RLfD to learn robotic motion skills. The DRL algorithm was initialized using demonstrations. The demonstrations took over the need for carefully engineered rewards, and reduced the exploration problem of classical RL approaches in these domains. In [88], the method was evaluated on a real robot, which learned to insert a flexible object into a rigid object.

In this thesis, GMM is used to model the human demonstrations, as it is a method that enables automatic extraction of trajectory constraints [90]. To enable adaptation to different environmental conditions, a GMM modification (m-GMM) algorithm is developed in this thesis. The m-GMM modifies the mean values of the learned GMM for the moving actions in order to adapt to the new pose of objects and to avoid collision with obstacles. GMR is then used to produce a smooth trajectory from the output of the m-GMM. The GMM/m-GMM/GMR method does not require demonstrations with different environmental conditions. Therefore, it functions with single and multiple demonstrations. Table 3 summarizes the robot learning at trajectory level methods, which are discussed in this section.

Table 3: Overview of Related Work in Low-Level Skill Learning at Trajectory Level.

Method for learning	Demos		Method for reproduction	Adaptation to environmental changes			Reference
	Single	Multiple		Different start pose	Different goal pose	Obstacle Avoidance	
DMP	√		DMP		√		Ijspeert et. al, 2002 [68], Schaal et. al, 2005 [69], Schaal, 2006 [70]
DMP	√		Modified DMP	√	√	√	Park et al., 2008 [71], Pastor, et al., 2009 [72]
DMP		√	MoMP	√	√		Muelling et al. 2013 [73]
DMP	√		Reinforcement Learning	√	√		Kober&Peters,2009 [40], Kormushev et al., 2010 [74], Deng et al., 2017 [75]
DMP/ GMM		√	GMR		√		Yin&Chen, 2014 [76], Chen et al., 2017 [77]
SEDS		√	SEDS	√			Khansari-Zadeh&Billard, 2011 [78]
HMM		√	trajectory generation (spline fitting/ interpolation)	√	√		Brand&Hertzmann, 2000 [79], Calinon et al., 2005 [80], Billard et al. 2006 [81]
HMM		√	GMR		√		Calinon, D'halluin el al., 2010 [54], Calinon, Sauser et al., 2010 [82]
TP-HSMM		√	GMR	√	√		Pignat&Calinon, 2017 [83]
GMM		√	GMR				Calinon el al, 2007 [84]
GMM		√	TP-GMM/GMR	√	√	√	Calinon, 2016 [85], Calinon, 2018 [86], Rozo et al., 2016 [87]
DRL + Demos	√	√	DRL	√	√		Vecerik et al., 2017 [88], Hester et al., 2018 [89]
GMM	√	√	m-GMM/GMR	√	√	√	Presented approach in this thesis (Section 3.4.2)

2.2.3 High-Level Task and Low-Level Skill Robot Learning

Current research in RLfD tends to focus either on continuous representations of the low-level motor control of robots or on discrete representations of the high-level task requirements [23] [25] [26]. It requires the development of algorithms capable of covering a wide variety of representations, from the continuous low-level motor control to sequence of actions, reasoning and symbolic representations of tasks [26]. Some researchers attempt to bridge the gap between symbolic task learning and skill learning at trajectory level by combining low-level and high-level RLfD approaches.

Growing Hierarchical Dynamic Bayesian Network (GHDBN) is a method proposed by Dindo & Schillaci in [91], which is used for the representation and reproduction of complex actions from demonstrations. The GHDBN is a two-level Hierarchical Dynamic Bayesian Network (HDBN) where one level describes the high level representation of the task and the other describes the low level behavior of the robot. Simulation results were presented on a complex task of the following skills: approach-object, grasp-and-dislocate-object, and hit-object.

Akgun & Thomaz, in [92], proposed a framework that learns the model of actions (skills) and goals (task). The teacher provided demonstrations including key-frames, which are a sparse (in time) set of ordered points. During the human demonstration, the teacher defined the key-frames by speech commands (e.g. go here). Two individual *Hidden Markov Models (HMMs)* were trained; the first HMM learned the sequence of actions to perform the task and the second HMM learned the trajectories. To evaluate the proposed method, two robotic tasks were selected – close the box and pour coffee beans from the cup to the bowl. In [93], Canal et al. modeled the demonstrated trajectories using the *Hidden Semi-Markov Models (HSMM)* and learns the sequence of actions using *Markov Decision Process (MDP)*. The proposed framework is designed for a shoe dressing task and the actions are predefined and based on user reactions. Evaluation results on a real robot are presented.

Hierarchical Deep Reinforcement Learning has been proposed by Yang et al. in [94], which learns skills and tasks simultaneously. The first level of hierarchy learned a particular basic skill, while the second level of hierarchy learned compound skills by

reusing basic skills in the first level of hierarchy to solve compound tasks. The method was evaluated with three different tasks (i.e. approaching object, approaching specific target, and doorway) in a simulated mobile robot.

In the presented thesis, a combination of high-level and low-level RLfD techniques is selected, similar to [93]. Interactive Reinforcement Learning (IRL) learns the sequence of actions to perform the task, enables interaction with the user (human-in-the-loop), and learns user's preference. On the other hand, the GMM learns the motor skills (low-level) and the m-GMM/GMR adapts them to new environmental conditions. Table 4 summarizes the related work, which is discussed in this section.

Table 4: Overview of Related Work in Low-Level and High-Level Robot Learning.

Symbolic Task Learning	**Skill Learning at Trajectory Level**	**HRI during execution**	**Robotic Application**	**Reference**
GHDBN (two layers)		No	Robotic arm: Task consists of the skills: approach-object, grasp-and-dislocate-object, and hit-object	Dindo & Schillaci, 2011 [91]
HMM	HMM	No	Humanoid robot: Close the box, Pour coffee beans from the cup to the bowl	Akgun&Thomaz, 2016 [92]
MDP	HSMM	Human-in-the-Loop	Robotic arm: Shoe dressing assistance	Canal et al., 2018 [93]
Hierarchical Deep Reinforcement Learning		No	Mobile robot: approaching object, approaching specific target, and doorway	Yang et al., 2018 [94]
IRL	GMM/m-GMM/GMR	Human-in-the-Loop, User Preferences	Dual-arm industrial robot: assembly, pins-into-holes. Robotic arm: drinking from a glass with a straw	Presented in this thesis (Section 3)

2.3 Robot Learning by Individuals with severe Motor Impairments

Research for assistive robotic manipulators for motor-impaired users is focused on development shared control paradigms for predefined tasks [95]. Robot learning from motor-impaired teachers is a new area of research [96], which could enable the motor-impaired users to teach desired tasks, without preprogramming. In [97], Argall highlighted how machine learning techniques could be used for learning of shared control for assistive robots.

In the work presented by Goil et al. in [98], the doorway navigation of a powered wheelchair was learned by human demos using the statistical machine learning approach Gaussian Mixture Model (GMM) and Gaussian Mixture Regression (GMR). However, the approach was only tested in a simulated environment and it was not evaluated in a real scenario with an extensive user study. A natural language interface was proposed by Broad et al. in [99], which enabled a user to provide corrections for assistive robotic manipulators. The user modified properties of how the robot achieved a goal on-the-fly. The user could correct properties such as speed, orientation, spatial constraint and grasping point. However, the robot actions that can be corrected were predefined and the tasks were preprogrammed. Similarly, in [100], Kuhner et al. proposed a system that the user can select predefined actions using a Brain-Computer-Interface (BCI).

In this thesis, the robot learning framework does not require predefined actions or pre-programmed tasks, so to provide more freedom to the user. The motor-impaired user controls the robot though the desired task and the robot learning framework learns the sequence of actions for a demonstrated manipulation task including the moving actions.

3. Robot Learning of Object Manipulation Tasks from Human Demonstrations

3.1 Overview of the Robot Learning Framework

The overall structure of the robot learning framework [1], [4] is shown in Figure 5. The framework enables robots to learn object manipulation tasks from human demonstrations. Most robot learning frameworks [22] [21] enable learning from multiple human teachers (demonstrators), so the learned task is independent of the experience and concentration of a single human teacher. The working hypothesis of multiple demonstrations, as described in [1], is that learning from a single demonstration has limitations, as the human teacher may make mistakes during the demonstration so that the robot could be vulnerable to these mistakes. Additionally, the human teacher has a lower precision compared to the robot and may perform unnecessary movements in attempts to be very precise in positioning the robot's end-effector. Furthermore, as the various human demonstrations lead to different demonstrated skills, an optimally learned skill could be the outcome of a combination of different demonstrations.

However, multiple demonstrations are not always possible. For example, it is very challenging and time consuming for a person with severe motor impairments to provide a single demonstration of a robot manipulation task via 'hands-free' interaction. The requirement of multiple demonstrations could burden the motor-impaired user physically

and/or mentally. The presented robot learning framework can rely on multiple demonstrations of a manipulation task. However, it can also work with a single demonstration, even if the demonstration may not be the most optimal in terms of the demonstrated path of the robot's end-effector and object grasping position. The different methods of Human-Robot Interaction (HRI), which enable human teachers to provide demonstrations, are presented in section 3.2. One assumption in this thesis is that the demonstration(-s) must complete the manipulation task successfully and the final goal of the task has to be achieved. For instance, in an assembly manipulation task, the final outcome of a demonstration must complete the desired assembled object. This thesis does not investigate robot learning from failed demonstrations [101].

As illustrated in Figure 5, the presented robot learning framework is organized in two main phases, the *learning phase (offline)* and the *working phase (online)* [1]. The presented robot learning framework has been developed in the open-source Robot Operating System (ROS) [102].

During the *learning phase* (section 3.3), all the actions needed to complete the object manipulation task are demonstrated once or several times by one or more human teachers. The robot learns the necessary sequence of actions for the demonstrated task, including the trajectories to manipulate the objects.

During the *working phase* (section 3.4), the robot is able to suggest the necessary sequence of actions to complete the task and to adapt the moving actions to environmental changes (including obstacle avoidance) in real-time. Moreover, the user can provide feedback regarding the suggested sequence of actions for the identified task and the robot learns the user's preferences online. In this thesis, the term 'user' is equivalent to terms human collaborator, human coworker, human teammate and end-user.

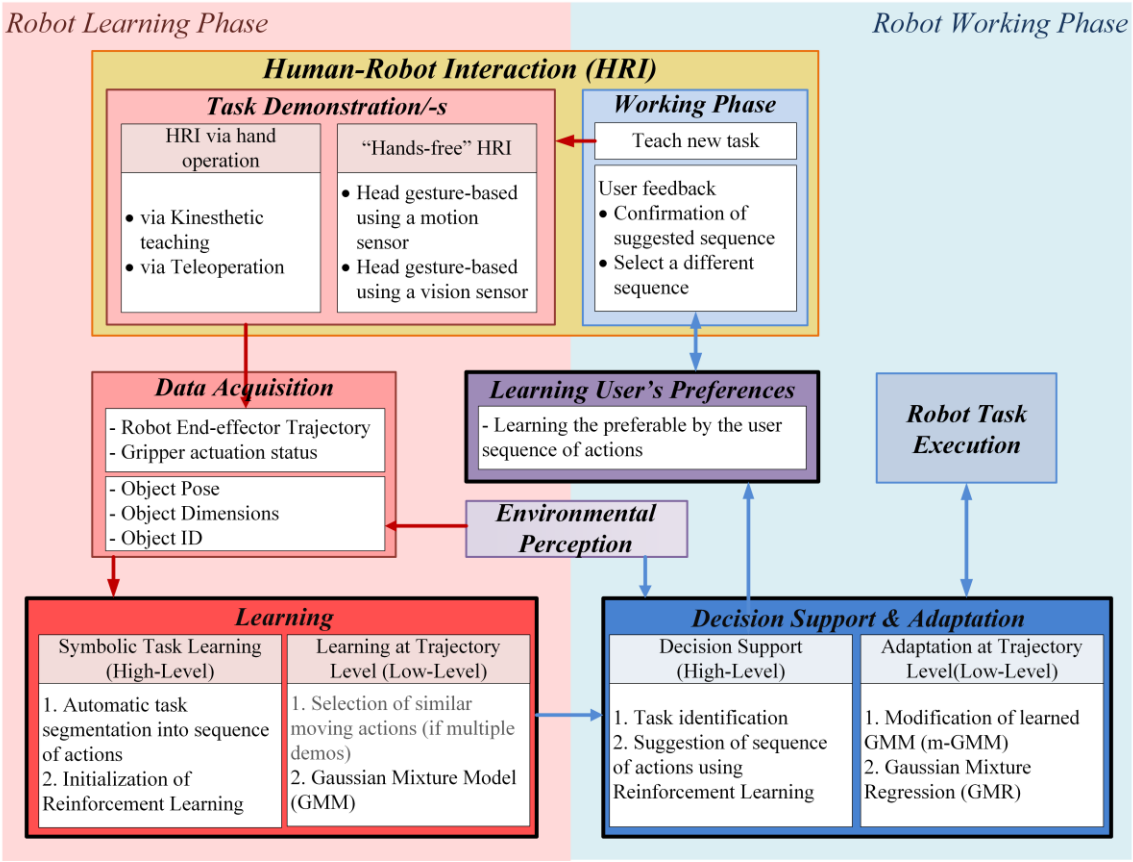


Figure 5: Overview of the developed robot learning framework (the main contributions of this thesis are: learning, decision support & adaptation and learning user's preferences blocks).

The *Environmental Perception module* is common in both phases (learning and working phase). A first version of this module (*v1*) was developed by Fang in [103] and by Yassin in [104]. It processes the Point Cloud Data from the vision-sensor, which observes the scene. Firstly, the state-of-the art YOLOv3 [105] object classifier is used to identify the object ID (for example: human, cup, pin, box) in the scene. Secondly, 3D-point cloud-based processing is performed to estimate the pose and the size of the identified objects. The output of this module is the object ID (for example: human, cup, pin, box), the object size and object pose with respect to the world coordinate system. A second version of this module (*v2*) was developed by Haseeb et al. [106] and [107], which uses YOLOv3 to recognize objects and to extract their bounding boxes, and *DisNet*, a multi-hidden-layer neural network, to estimate the position of object with respect to a monocular camera (RGB). The development of the *Environmental Perception module* is not part of this thesis and the output of the *Environmental Perception module* is assumed to be accurate.

3.2 Human-Robot Interaction for Robot Learning

In this section, several HRI methods for robot learning are presented, which are classified into two groups; 'hand-operated' and 'hands-free' HRI. The hand-operated HRI is operated by the user's hands by one of two methods, kinesthetic teaching (section 3.2.1.1) and teleoperation (section 3.2.1.2). The hands-free HRI is based on head gesture-based recognition and there are two approaches – using motion (section 3.2.2.1) or vision sensor (section 3.2.2.2). The 'hands-free' HRI can be used for tetraplegic users or when the user's hands are occupied with another task. During the demonstrations of a manipulation task, the initial scene is the same for every demonstration.

3.2.1 'Hand-Operated' Human-Robot Interaction for Robot Learning

3.2.1.1 Human-Robot Interaction via Kinesthetic Teaching

A widely-used method to provide demonstrations for robot learning is kinesthetic teaching. In this method, the user controls the robot by physically grasping the robot's end-effector and guiding it through the task. The robot's joints are set in gravity compensation mode [108], which allows the user to maneuver the robot's end-effector. In the presented work, the actuation and de-actuation of the grippers (robot's end-effectors) are also controlled by the user during the demonstrations.

3.2.1.2 Human-Robot Interaction via Teleoperation

Another method for able-bodied users to interact with the robot and provide the demonstrations is to control the robot via a gamepad [109], which is shown in Figure 6. The different buttons on the gamepad provide the following functionalities [6]: selection of robot arm (left and right) in case of dual-arm robots, the grippers' actuation and de-actuation, and control of one out of 6 dimensions at the time, specifically translation (*x, y, z*-axis) or rotation (roll, pitch, yaw) with respect to the world coordinate system.

Robot control via gamepad is physically less demanding for the users in comparison to kinesthetic teaching and it is easier as the user needs to directly control the translation

and rotation of the robot's end-effector without worrying about the joint configuration. Moreover, some of the users are already familiar with the use of gamepads.

Figure 6: Gamepad.

3.2.2 'Hands-free' Human-Robot Interaction for Robot Learning

Head gesture-based HRI is one approach that enables 'hands-free' interaction. Different sensors, motion (section 3.2.2.1) and vision sensors (section 3.2.2.2) are used to measure the head movement. One popular approach for head gesture recognition is to apply classifiers [110] [111], such as Support Vector Machines in [112] [113] and Hidden Markov Models in [114] [115]. In the presented thesis, Support Vector Machine (SVM) [116] is selected as a classification technique to recognize the head gestures [4] [5].

A flowchart of the robot control via head gesture-based interface is illustrated in Figure 7, and it is similar in both approaches (motion and vision sensor). Firstly, there is the training phase of the SVM, which is indicated with the red arrows in Figure 7. Participants performed the head gestures while wearing the sensor (motion or vision) and the sensor data was collected for each gesture. Features were extracted from the collected labeled data and the SVM was trained.

The robot control via head gesture-based interface in real-time is shown in Figure 7 with the blue arrows. After the data acquisition from the sensor and after the subsequent feature extraction, the performed head gesture is recognized with the trained SVM model. The user utilizes the performed and recognized head gestures to navigate through the state machine[4] of the robot. Examples of the state machine of robots are shown in Figure 9 and Figure 13. The user is able to select a state, such as to control one of the rotation

[4] State machine (or finite state machine) is a model of a computation system, consisting of states, possible inputs and a rule to map each state to another (or to itself), for any of the possible inputs [156]

angles (pitch, yaw, roll) or the translation axes (*x-, y-, z-* axis) of the robot's end-effector, as well as to control the gripper's actuation (Open/ Close) or to select different robotic arms, in case of dual-arm robots. The states where the user can control a dimension of the robot's end-effector are called *control states*. The audio feedback always assists the user by announcing the selected state. With the audio assistance, the user can control the robot by directly looking at it.

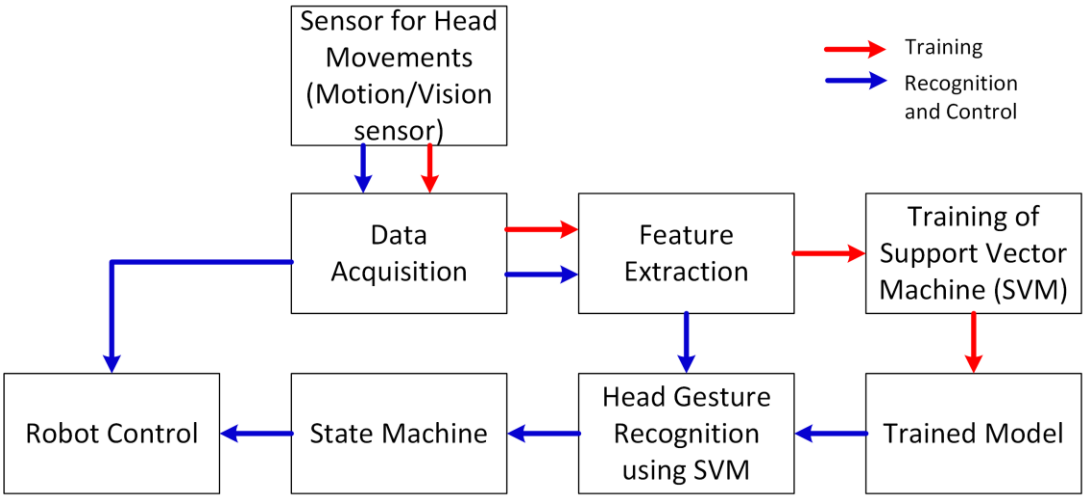


Figure 7: Overview of the head gesture-based HRI. (adapted from [4])

Four head gestures that are needed to navigate through the state machine are explained in Table 5. The gestures are shown in Figure 9 and Figure 13. An important note is that the gestures were selected based on the neck motion limitations of a tetraplegic user suffering from multiple sclerosis.

Table 5: Explanation of Head Gestures.

Name of the Gesture	Explanation
Up-Gesture (UG)	Up nod of the head
Down-Gesture (DG)	Down nod of the head
Left-Gesture (LG)	Left turn of the head
Right-Gesture (RG)	Right turn of the head

After the selection of a control state, the left and right turn of the head is mapped to the movement of the robot in negative and positive directions of the selected dimension (robot control). The more the user's head turns, the more the robot moves along the selected dimension. The Down-Gesture (DG) is used to stop the direct robot control and to return back to the state machine. As the range of the head motions may differ for each

user, a calibration procedure is done before the user starts interacting with the robot. During calibration, the user turns the head to the left and to the right, while the system identifies the range of the head motion.

More details about the sensor setup, the feature extraction, the accuracy of the SVM, the state machine and the robot control are given in sections 3.2.2.1 and 3.2.2.2 for head gesture-based HRI using a motion and a vision sensor, respectively.

3.2.2.1 Head Gesture-based Human-Robot Interaction Using a Motion Sensor

An approach for 'hands-free' HRI is based on head gesture recognition using a motion sensor and has been presented by Haseeb, Kyrarini et al. in [5]. The myAHRS+ sensor [117] from WITHROBOT is selected as a motion sensor and it consists of a 3-axis accelerometer, gyroscope and geomagnetic sensor. Moreover, the motion sensor has on-board data fusion to estimate the rotation in Euler angles and quaternions. The sensor is mounted on a headband, which is worn by the user, as shown in Figure 8a. The orientation of the sensor is shown in Figure 8b.

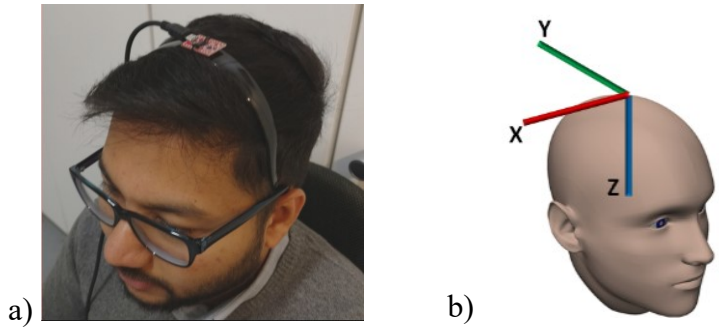


Figure 8: a) The motion sensor is worn by the user, b) Orientation of the motion sensor [5].

The head gesture data for training the SVM were collected from one tetraplegic and 12 able-bodied participants. The participants wore the motion sensor on the top of their heads and they were asked to repeat each gesture 10 times. The recorded data consisted of 3-dimensional acceleration and 3-dimensional angular velocity, which were normalized into a range from [0,1]. The selected features, such as minimum and maximum values of the acceleration and angular velocity were used to train the SVM. The accuracy of the head gesture classification was 98.42% for the able-bodied participants (using k-fold cross-validation), while for the tetraplegic participant it was

89.72% (the data of the tetraplegic participant was included in the training and testing dataset) [5]. The lower accuracy for the tetraplegic user was due to the slower speed and the range of performing the head gestures was smaller than the range of able-bodied users.

A user using these head gestures is able to navigate through the state machine of a dual-arm industrial robot with multiple grippers (more information about the robotic platform are given in section 4.1). The state machine of the dual-arm robot is shown in Figure 9.

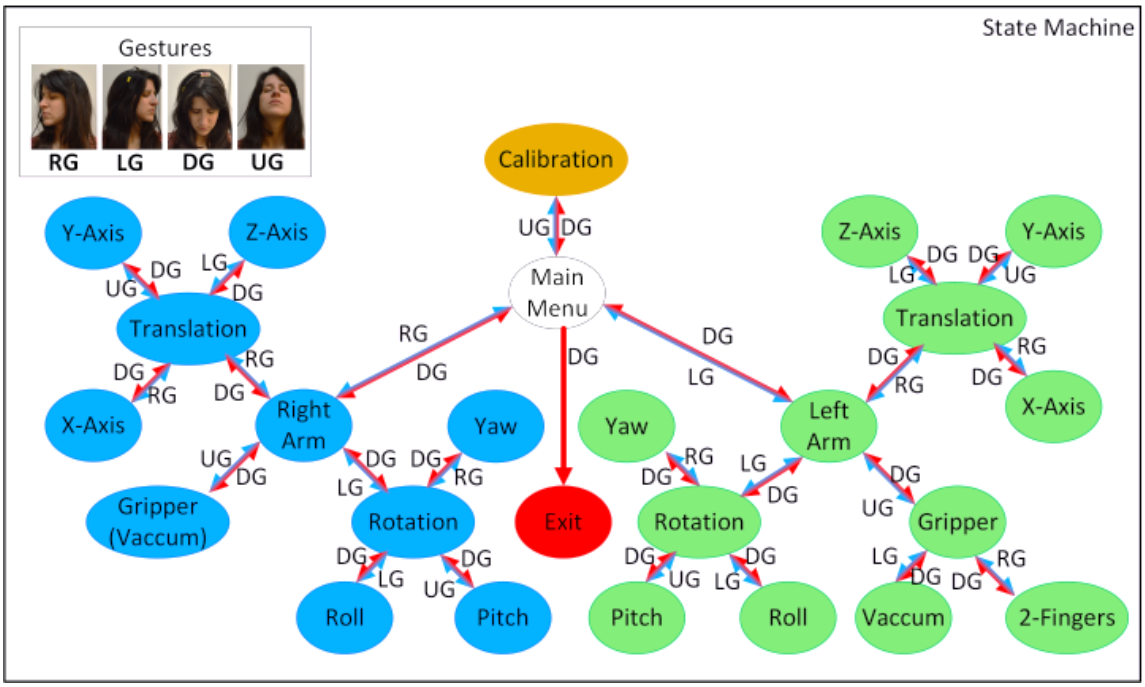


Figure 9: State machine of a dual-arm industrial robot with multiple grippers [5].

After the selection of a control state, the velocity of the robotic arm's end-effector in a selected dimension is controlled by the rotation of the user's head around the z-axis with respect to the neutral position of the head. The more the user's head turns, the faster the robot moves along the selected dimension. During calibration, the maximum velocity along the positive axis is defined as the range of motion of the head in the right direction and the maximum velocity along the negative axis is defined as the range of motion in the left direction of the head as shown in Figure 10.

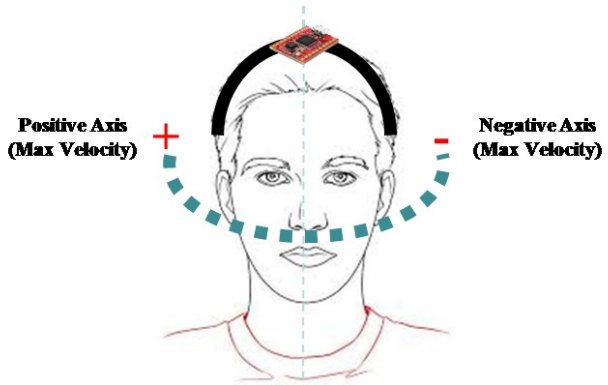


Figure 10: Mapping between velocity of robotic arm's end-effector and range of head motion [5].

3.2.2.2 Head Gesture-based Human-Robot Interaction Using a Vision Sensor

The head gesture-based HRI consists of a hat equipped with a low-cost miniature web-camera [118] (vision sensor), as shown in Figure 11 and has been presented by Kyrarini et al. in [4]. The camera points towards the body of the user sitting still in a wheelchair. The working principle of this approach is that the human body is a static environment and the changes in the scene viewed by the camera are due to the head motion. These detected changes define head gestures.

Figure 11: The vision sensor is worn by the user suffering from tetraplegia. (adapted from [4])

The first step for head gesture recognition is to collect the labeled data in order to train the SVM. Ten able-bodied participants were asked to perform the four gestures (up/down/left/right) by repeating each gesture ten times, while wearing the hat with the camera, during which data (video) was recorded. Each performed gesture is stored as image frames (video).

First, the Shi-Tomasi detector [119] is used for detection of corner points from the initial frame of the video. Subsequently, the optical flow vectors [120] of the tracked corner points are calculated using the Lucas-Kanade method [121] through subsequent continuous frames. An example of the detection of corner points and optical flow is shown in Figure 12. Each gesture is represented by one normalized optical flow vector used as a feature to train the SVM classifier offline in a supervised manner. The accuracy of the head gesture recognition for seven participants (6 able-bodied and 1 tetraplegic) who did not belong to the training set was 97.61% [4]. Moreover, the accuracy of the head gesture recognition for the tetraplegic participant was 95.83% [4].

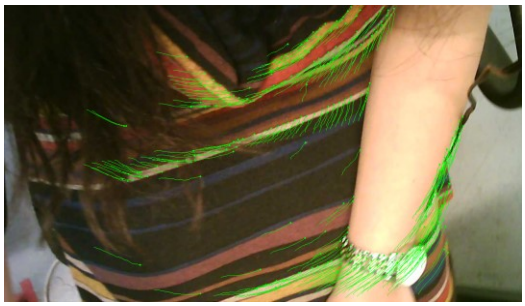
a)  b) 

Figure 12: Example of a) detection of corner points at initial frame of the video and b) optical flow while performing the right head gesture.

The user utilizes the performed and recognized head gestures to navigate through the state machine of an assistive robotic manipulator equipped with a gripper (more information about the robotic platform are given in section 5.1), which is shown in Figure 13. After the selection of a control state, the range of head motion to the left and right is mapped to the movement of the robot in negative and positive directions of the selected dimension in position control mode. The more the user's head is turned, the more discrete steps the robot takes.

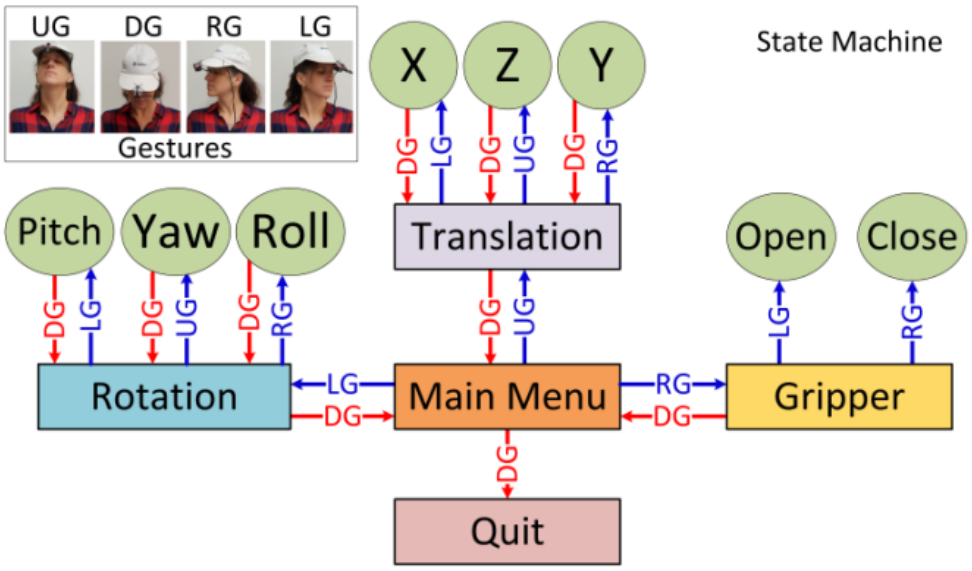


Figure 13: The state machine of an assistive robotic manipulator [4].

3.3 Robot Learning Phase (Offline)

The *offline robot learning phase* [1] consists of the following two main modules.

- *Data Acquisition module:* This module records the trajectory of the robot's end-effector (end-effector pose with respect to the world coordinate system and time) as well as the actuation status of the gripper during the human demonstrations. The recorded data are stored in the robot database. Each demonstration presents the complete object manipulation task, which is demonstrated by a user. The trajectory consists of data-points *{s, t}*, where *s* refers to the 7-dimensional spatial variables *{x, y, z, qx, qy, qz, qw*[5]*}* and *t* is the temporal variable. The sampling frequency is defined as 100Hz for all the demonstrations. Additionally, this module stores the outputs of the *Environmental Perception module* in the robot database, which include the object ID, the object size and object pose with respect to the world coordinate system.
- *Learning module:* This module enables the robot to learn the sequence of actions needed to complete the demonstrated object manipulation task without any pre-programming by the user. The *learning module* is organized into two layers.

[5] *qx, qy, qz, qw:* quaternions

— *Symbolic Task Learning (High-level Learning),* consisting of the following steps: Automatic Task Segmentation into Sequence of Actions (ATSSA) and initialization of Reinforcement Learning from human demonstrations. More details are given in section 3.3.1.

— *Learning at Trajectory Level (Low-level Learning),* consisting of two steps resulting in the learned end-effector trajectory. These steps are the selection of similar demonstrations and the Gaussian Mixture Model (GMM). A lookup table is also generated that associates the high-level actions with the trajectories. More details are given in section 3.3.2.

The outcomes of the *Learning module* are stored in the database and they comprise as follows:

1. the initialized Q-Table for Reinforcement Learning,
2. the object ID, size and pose for the involved objects,
3. the learned Gaussian Mixture model (GMM) of the demonstrated end-effector's trajectories, and
4. the lookup table that associates the high-level actions with the moving actions.

3.3.1 Symbolic Task Learning (High-level)

3.3.1.1 Automatic Task Segmentation into Sequence of Actions (ATSSA)

The ATSSA sub-module is responsible for generating the sequence of actions for the demonstrated manipulation task. The trajectory of the robot's end-effector, the actuation status of the gripper and the outputs of the *Environmental Perception module* for each demonstration are used as input to the ATSSA. The ATSSA first identifies when the robot is in the 'home' pose (home action) and detects the gripper actions. There are two types of gripper actions: the gripper is either actuated (close action) or not-actuated (open action). The gripper close action is the event when the gripper state changes from 'open' to 'close', and the gripper open action is when it changes from 'close' to 'open'.

Subsequently, the ATSSA associates each gripper close action with the information of the manipulated object (provided by the *Environmental Perception module*) as well as

each gripper open action with the information of the object, which is close to the placement pose of the manipulated object. Additionally, if an object manipulation does not start or end in the 'home' pose of the robot, the method detects which object is closer to the end-effector at the start or the end of the end-effector's trajectory. In particular, if the robot is not in the 'home' pose at the start/end of an object manipulation, the robot action 'Start/End' is assigned by ATSSA, respectively. The output of this method is a sequence of high-level actions with pairs of robot action a^r ($a^r \in \{$*Start, End, Close, Open, Home*$\}$) and object information a^{obj} (ID, size, pose) for each demonstration.

For example, in the case of a simple manipulation task, the user guides the robot to pick up a small box and to place it on top of a big box. The demonstration starts and ends in the robot home pose. The output of the ATSSA is:

1. *Home – Home*[6],
2. *Close – Small box info*,
3. *Open – Big box info*,
4. *Home – Home*.

To provide another example, in the case of the demonstration of a drinking task, the user controls the robot from a pose near the table, to pick up a cup and to deliver it close to their mouth. The output of the ATSSA is:

1. *Start – Table info*,
2. *Close – Cup info*,
3. *End – Person info*.

For dual-arm robotic platforms with multiple grippers, the robot actions can be updated to include information about which arm and which gripper is used. Real-world examples can be found in chapters 4 and 5.

3.3.1.2 Initialization of Reinforcement Learning from Human Demonstrations

In the presented work, Reinforcement Learning (RL) is selected to enable the learning of a sequence of actions. Schaal [122] proposed RL from human demonstrations, which is the initialization of the value-function (Q-Table) of RL by using demonstrations. The

[6] When robot is in the 'home' pose, no object is considered.

demonstrations are assumed to be a passive learning experience for the RL agent. The Q-Table is initialized by the human demonstration of the manipulation tasks and converges to a final value quickly. The output of the ATSSA sub-module for all the demonstrations encompasses the actions A of the Q-Table. The states are g =<1, …, G>[7], where G is the total number of performed actions (total number of states). The reward $\hat{R}(g_\tau, a_\tau, g_{\tau+1})$ from the human demonstrations is calculated by the equation (3.1):

$$\hat{R}(g_\tau, a_\tau, g_{\tau+1}) = \frac{k}{D} \tag{3.1}$$

where g_τ is the current state in time τ, a_τ is the action in time τ, $g_{\tau+1}$ is the new state in time τ+1, D is the total number of human demonstrations, and k is the number of human demonstrations at which action a_τ leads to state $g_{\tau+1}$. The equation (3.2) is used for the initialization of the values of the Q-Table.

$$Q(g_\tau, a_\tau) = \hat{R}(g_\tau, a_\tau, g_{\tau+1}) \tag{3.2}$$

For example, in the case of an assembly manipulation task in which 2 pins (one small and one big) are inserted into the two holes of a holder, there are D number of demonstrations, and in D_1 demonstrations the teachers manipulate the small pin first and in D_2 demonstrations they manipulate the big pin first, where $D_1, D_2 < D$ and $D_1 + D_2 = D$. The output of the ATSSA is shown in Table 6.

Table 6: Output of the ATSSA in case of an assembly manipulation of inserting 2 pins (one small and one big) into two holes of a holder.

τ	D_1 demonstrations	D_2 demonstrations
1	Home – Home	Home – Home
2	Close – Small Pin info	Close – Big Pin info
3	Open – Holder info	Open – Holder info
4	Close – Big Pin info	Close – Small Pin info
5	Open – Holder info	Open – Holder info
6	Home – Home	Home – Home

The actions are *A=< Home – Home*, *Close – Small Pin info*, *Open – Holder info*, *Close – Big Pin info>* and the states are g =*<1,…,6>*. Based on equations (3.1) and (3.2),

[7] In RL the state usually is denoted as *s*, but in this thesis g is used. The reason is to avoid confusion with *s* as 7-dimensional spatial variables.

the initialization of the Q-table is shown in Table 7 for the assembly manipulation of inserting 2 pins into the two holes of a holder. The use of the initialized Q-Table in the online robot working phase is explained in section 3.4.1.

Table 7: Initialization of the Q-Table for an assembly manipulation task of inserting 2 pins (one small and one big) into the two holes of a holder.

	Actions *a*			
States *g*	*Home – Home*	*Close – Small Pin info*	*Open – Holder info*	*Close – Big Pin info*
1	$(D_1 + D_2)/D$	0	0	0
2	0	D_1/D	0	D_2/D
3	0	0	$(D_1 + D_2)/D$	0
4	0	D_2/D	0	D_1/D
5	0	0	$(D_1 + D_2)/D$	0
6	$(D_1 + D_2)/D$	0	0	0

3.3.2 Skill Learning at trajectory level (Low-level)

3.3.2.1 Selection of similar demonstrations

The human demonstrations that are given as input to the robot learning framework are crucial, as the learned robot behavior is based on the demonstrations. During multiple human demonstrations of a task, there are two main problems: the different speed of each demonstration and the variety of demonstrated trajectories [2] [3] [6].

Dynamic Time Warping (DTW) [123] is a method that is used to find an optimal alignment between two time-series which may vary in speed and time. DTW is a widely used method to align human demonstrations that differ in speed and time [124] [125] [2]. Moreover, DTW measures the similarity between two time series and it is proposed as a solution for finding similar demonstrations based on the demonstrated trajectories [2] [3] [84]. However, one disadvantage of DTW is the high degree of complexity of the algorithm, which is time consuming [126].

A novel method for alignment and selection of similar demonstrations without the use of DTW is presented in this section and has been presented by Kyrarini & Gräser in [6].

More precisely, the presented algorithm is able to align and select similar trajectories. The presented method uses the complete trajectories from the *D* demonstrations as input, where *D* is the total number of demonstrations. A demonstrated trajectory consists of data-points *{s, t}*, where *s* refers to the 7-dimensional spatial variables *{x, y, z, qx, qy, qz, qw}* and *t* is the temporal variable (sample). Additionally, the temporal variable t_g is also considered input, which is the time when the robot takes an action to go to state g, e.g. when the robot picks up an object.

- *Step 1: Splitting trajectories into sub-trajectories*

The trajectory of each demonstration is split (segmented) based on the temporal variable t_g into $G-1$ sub-trajectories, where G is the total number of states (section 3.3.1.1). For example, if there are 4 states (G) during the demonstrated task, the complete trajectory will be split in 3 sub-trajectories ($G-1$). By splitting the trajectories into sub-trajectories, the different demonstrations are spatially aligned on the Important Points (IP) for the demonstrated task. IP in the object manipulation tasks are the points for grasping (picking up) and releasing (placing) the objects, which require precision. A sub-trajectory is denoted as $P_{d,g}$ where $d = 1, \dots, D,\ g = 1, \dots, (G-1)$. All the g-th sub-trajectories of the *D* demonstrations are organized in a group (g). An example of splitting each demonstrated trajectory into 3 sub-trajectories is shown in Figure 14. In the case of a single demonstration of a task (one-shot robot learning), the rest of the steps (steps 2 – 4) are skipped.

- *Step 2: Resampling the sub-trajectories in a group (g)*

Within a group (g) of sub-trajectories, the total number of samples (time duration) may vary due to different speeds of performing the demonstrations. The polyline simplification algorithm by Ramer-Douglas-Peucker (RDP) [127], [128] is used to provide sub-trajectories of the same total number of samples. As described by Kyrarini et al. in [7] and [3], polyline simplification methods are used to automatically select the important 'key-points' of a trajectory. The RDP algorithm is explained in Appendix B.

The inputs to the RDP algorithm are the total desired number of samples and the sub-trajectories $P_{d,g}$ of the g-th group. The total desired number of samples for each group is

equal to the smaller total number of samples among the sub-trajectories of the group. The outputs are the RDP sub-trajectories $\hat{P}_{d,g}$ of the same total number of samples $\hat{T}_g$.

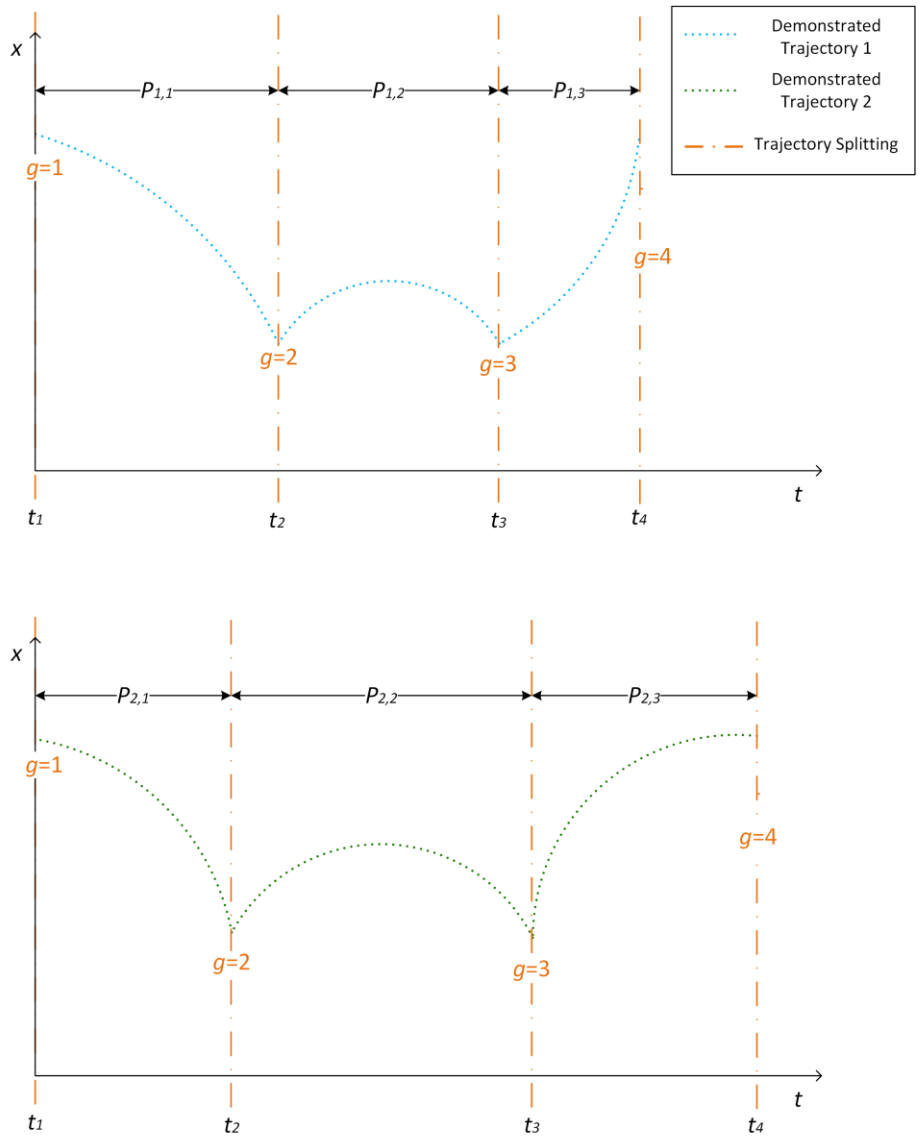


Figure 14: Example of splitting two demonstrated trajectories into sub-trajectories for one spatial variable.

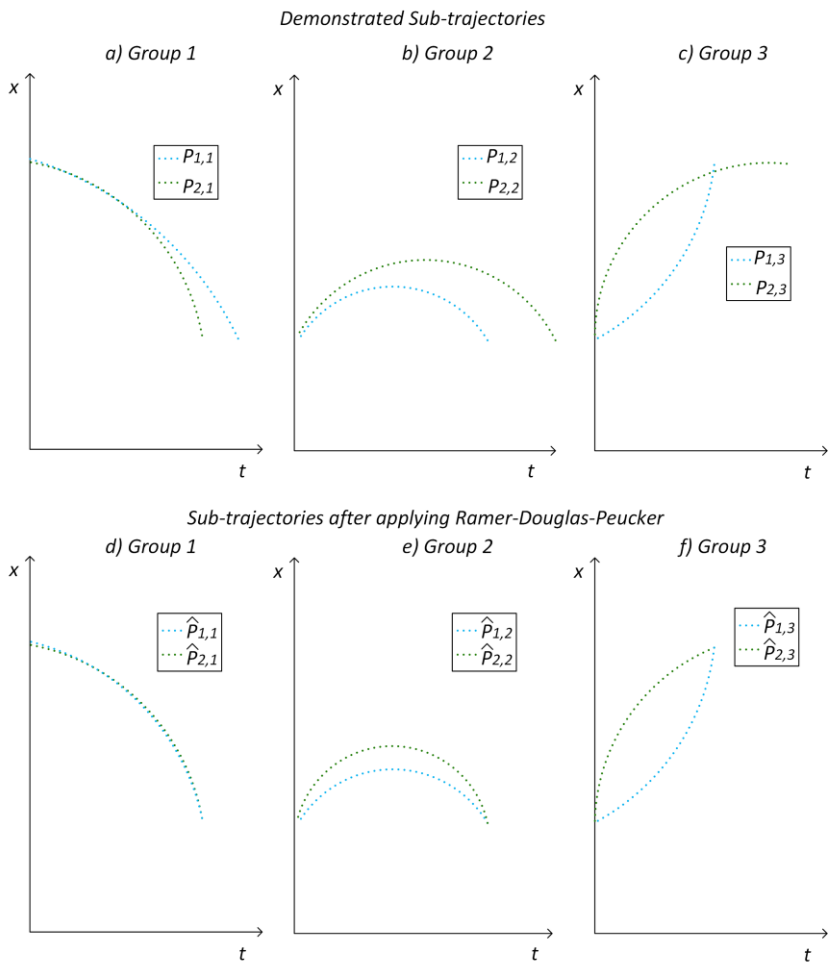


Figure 15: Example of the split sub-trajectories organized in groups (a - c) and the sub-trajectories after applying Ramer-Douglas-Peucker (d - f) for one spatial variable.

➢ *Step 3: Calculation of dissimilarity between the sub-trajectories of the g-th group*

The dissimilarity matrix *dsim* between the sub-trajectories of the g-th group is calculated by the equation (3.3):

$$dsim_g(i,j) = w_1 \cdot distP\left(\hat{P}_{i,g}, \hat{P}_{j,g}\right) + w_2 \cdot distIP\left(\hat{P}_{i,g}, \hat{P}_{j,g}\right) \tag{3.3}$$

where $i,j = 1,\dots,D$. $distP$ is the sum of the seven-dimensional Manhattan point-to-point distance between the points of two sub-trajectories $\hat{P}_{i,g}$ and $\hat{P}_{j,g}$. $distIP$ is the seven-dimensional Manhattan distance between the IP of the two sub-trajectories. w_1 and w_2 are the weights that satisfy the following equations: $0 \leq w_1 \leq 1, w_2 = 1 - w_1$, $w_1 < w_2$. For all the data-points the dimensions *{x, y, z}* are normalized in the range of [-1, 1]. The higher the value of the $dsim_g(i,j)$ for the sub-trajectories of the i-th and j-th

demonstrations, the less similar (in the sense of the demonstrated path and IP) the demonstrations are. Further, the dissimilarity vector $dsimV$ of the sub-trajectories of the g-th group is calculated by the equation (3.4):

$$dsimV_g(i) = \sum_{j=1}^{D} dsim_g(i,j), \forall i \epsilon \{1,..,D\} \tag{3.4}$$

Manhattan distance between two data-points:

The Manhattan distance $dist$ between two data-points $\beta_1(s)$, $\beta_2(s)$, where s={*x, y, z, qx, qy, qz, qw*}, is calculated by the equation: $dist = \sum_{s=1}^{7} |\beta_1(s) - \beta_2(s)|$. One advantage of Manhattan Distance is that it is fast to compute [129].

Manhattan distance between two trajectories:

A trajectory consists of data-points $\beta(s,t)$, where t is the temporal variable (sample). The Manhattan distance $distP$ between two trajectories with data-points $\beta_1(s,t)$ and $\beta_2(s,t)$ is calculated by the equation: $distP = \sum_{t=1}^{T}(\sum_{s=1}^{7}|\beta_1(s,t) - \beta_2(s,t)|)$, where T is the total number of samples.

- *Step 4: Selection of sub-trajectories for the g-th group*

For the g-th group, the sub-trajectory with the smallest value in the $dsimV_g$ is selected as the 'reference' *r*, since this sub-trajectory has the smallest difference between all the demonstrated sub-trajectories. After the selection of the reference sub-trajectory *r*, it is necessary to identify the sub-trajectories that are similar to the *r* [3]. The sub-trajectories that satisfy the inequality (3.5) are similar to the *r*, and they can be further used as input to the Gaussian Mixture Model (GMM).

$$dsim_g(r,j) \leq c, \forall j \epsilon \{1,..,D\} \tag{3.5}$$

where c is a predefined threshold value.

The aligned and selected sub-trajectories are denoted as $\hat{P}_{d',g}$, where $d' = 1, \dots, D'_g$ and D'_g is the total number of the selected sub-trajectories for the g group. Steps 2 – 4 are repeated for all the $(G-1)$ groups. Each group of sub-trajectories represents a

moving action between two high-level actions. A lookup table stores the association between the high-level actions and the groups of sub-trajectories.

3.3.2.2 Gaussian Mixture Model (GMM)

The input to Gaussian Mixture Model (GMM) is the above described result on the selection of similar sub-trajectories. The GMM is used to extract constraints of the aligned trajectories [84].

The selected and aligned sub-trajectories are fed to the GMM in order to build the probabilistic model of the data. The sub-trajectories of a group consists of data-points $\beta_\gamma = \{\beta_s, \beta_t\}$, where $\beta_s \in R^{dim}$, dim=7, $\beta_t \in R$, $\gamma = 1, \dots, \Gamma$ and $\Gamma = D'_g \cdot \hat{T}_g$. In the learning phase, the model is created with a number N of Gaussians. Each Gaussian consists of the following parameters: mean vector, covariance matrix and the prior probability [130]. Each Gaussian has a dimensionality 8 (dim+1). The probability density function $p(\beta_\gamma)$ for a mixture of N Gaussians is calculated based on the equation (3.6) [130]:

$$p(\beta_\gamma) = \sum_{n=1}^{N} p(n) \cdot p(n|\beta_\gamma) \tag{3.6}$$

where $p(n)$ is a prior and $p(n|\beta_\gamma)$ is a conditional probability density function. They are defined by the equations (3.7) and (3.8) for a mixture of N Gaussian distributions of dimensionality 8 (7 spatial and 1 temporal dimensions) [130]:

$$p(n) = \pi_n \tag{3.7}$$

$$p(n|\beta_\gamma) = \frac{1}{\sqrt{(2\pi)^{(dim+1)}|\Sigma_n|}} e^{-\frac{1}{2}\left[(\beta_\gamma-\mu_n)^{\mathrm{T}}\Sigma_n{}^{-1}(\beta_\gamma-\mu_n)\right]} = \aleph(\beta_\gamma; \mu_n, \Sigma_n) \tag{3.8}$$

where π_n are the prior probabilities, $\mu_n = \{\mu_{n,t}, \mu_{n,s}\}$ are the mean vectors and $\Sigma_n = \begin{pmatrix} \Sigma_{n,t} & \Sigma_{n,ts} \\ \Sigma_{n,st} & \Sigma_{n,s} \end{pmatrix}$ are the covariance matrices of the GMM. A Gaussian distribution of center μ and covariance matrix Σ is denoted as $\aleph(\mu, \Sigma)$ and the probability of a data point

β with respect to this Gaussian distribution as $\aleph(\beta; \mu, \Sigma)$. The parameters (prior, mean and covariance) of the GMM are estimated by the Expectation-Maximization (EM) algorithm [131]. The Figure 16 illustrates the probabilistic model of a dataset through GMM. For each group of selected sub-trajectories, a GMM model is learned and the GMM parameters are stored.

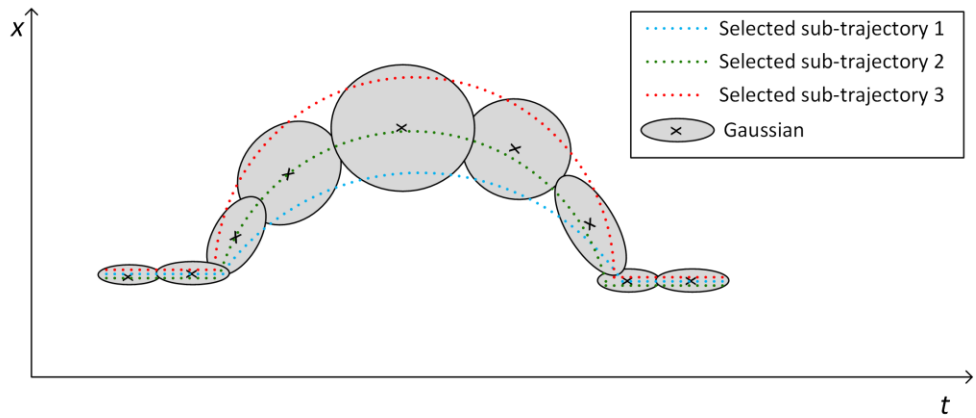


Figure 16: The learned GMM for one spatial dimension using three selected sub-trajectories as input.

- **Estimation of GMM parameters using Expectation-Maximization algorithm**

The EM algorithm is a simple local search method which guarantees monotone increase of the likelihood during optimization [130]. The $p_{n,\gamma}$ is the posterior probability and E_n is the sum of the posterior probabilities. The parameters π_n, μ_n, Σ_n, E_n of the GMM are initialized by K-means [132] and then are estimated iteratively until convergence, as follows.

➢ *E-step:*

$$p_{n,\gamma}^{(v+1)} = \frac{\pi_n^{(v)} \cdot \aleph(\beta_\gamma; \mu_n^{(v)}, \Sigma_n^{(v)})}{\sum_{i=1}^{N} \pi_i^{(v)} \cdot \aleph(\beta_\gamma; \mu_i^{(v)}, \Sigma_i^{(v)})},$$

$$E_n^{(v+1)} = \sum_{\gamma=1}^{\Gamma} p_{n,\gamma}^{(v+1)}.$$

- *M-step:*

$$\pi_n^{(v+1)} = \frac{E_n^{(v+1)}}{\Gamma},$$

$$\mu_n^{(v+1)} = \frac{\sum_{\gamma=1}^{\Gamma} p_{n,\gamma}^{(v+1)} \cdot \beta_\gamma}{E_n^{(v+1)}},$$

$$\Sigma_n^{(v+1)} = \frac{\sum_{\gamma=1}^{\Gamma} p_{n,\gamma}^{(v+1)} \cdot (\beta_\gamma - \mu_n^{(v+1)}) \cdot (\beta_\gamma - \mu_n^{(v+1)})^{\mathrm{T}}}{E_n^{(v+1)}}.$$

The iterations v stop when $\frac{L(\gamma,\Theta)^{(v+1)}}{L(\gamma,\Theta)^{(v)}} < c_1$, where $L(\gamma,\Theta) = \sum_{\gamma=1}^{\Gamma} \log\left(p\left(\beta_\gamma|\Theta\right)\right)$ is the log-likelihood, $\Theta = \{\pi_n, \mu_n, \Sigma_n\}_{n=1}^{N}$ is the model of the GMM and $p\left(\beta_\gamma|\Theta\right)$ is the probability that the data-point β_γ is generated by the model Θ. The threshold c_1 is set to 0.01, which is suggested in [130].

- **Estimation of optimal number using Bayesian Information Criterion**

The optimal number N of the GMM components (Gaussians) is estimated using the Bayesian Information Criterion (BIC) [133], which is commonly used criterion [134]. The BIC score is defined in the equation (3.9) [130] and it is calculated for a range of N' values, where $N'$is a number of Gaussians.

$$BIC = -L(\gamma,\Theta) + \frac{n_p}{2} \cdot \log(\Gamma) \tag{3.9}$$

where $L(\gamma,\Theta)$ is the log-likelihood of the GMM model, n_p is the number of independent parameters required for a GMM of N' components and is calculated based on equation (3.10) [130] for a GMM with full covariance matrix and Γ is the number of data-points with dimensionality 8 (*dim*+1).

$$n_p = (N'-1) + N' \cdot \left[(dim+1) + \frac{1}{2} \cdot (dim+1) \cdot (dim+2)\right] \tag{3.10}$$

The first term of the equation (3.9) estimates how well the data are obtained given the model Θ, and the second term is a penalty factor with a goal to minimize the number of independent parameters [130] [135]. Henceforth, the BIC score of GMMs with different number of Gaussians N' is estimated and the optimal number of Gaussians N is the one with the minimum BIC score.

3.4 Robot Working Phase (Online)

During the *robot working phase*, the robot performs the learned task in real-time, collaborating with the user to complete the task. The *robot working phase* [1] consists of the following three main modules.

- *Decision Support and Adaptation (DSA) module:* This module enables the robot to identify the sequence of actions needed to complete the learned object manipulation task. The *DSA module* is organized into two layers.
  - *Decision Support (High-level)* identifies the high-level actions. It consists of the following steps: task identification and suggestion of sequence of actions using Reinforcement Learning. More details are given in section 3.4.1.
  - *Adaptation at Trajectory Level (Low-level Learning)* consists of a novel algorithm for modification of the learned GMM for the real-time adaptation of the learned trajectories to the new environmental conditions. The new environmental conditions could arise due to changes in position and orientation of the objects, as well as due to obstacles. The modification of the learned GMM is followed by the Gaussian Mixture Regression (GMR) method, which generates the adapted trajectory for the robot. The GMR trajectory enables the robot to complete the manipulation task. More details are given in section 3.4.2.
- *Robot Learning User's Preference module:* This module enables the robot to learn the preferable sequence of actions for each user in real-time. The method is based on the RL algorithm. More details are given in section 3.4.3.

- *Robot Task Execution module:* This module enables the robot to execute the manipulation task. A virtual environment has been developed using the ROS-based tool rviz [136] to illustrate the environmental awareness and the execution of the learned task by the robot. For safety reasons, the user first observes what the robot intends to do (moving actions, grasping or releasing an object) in the virtual environment. Then, if safety criteria are satisfied, the user confirms that the real robot can perform the visualized sequence of actions.

3.4.1 Decision Support (High-level)

The Decision Support (DS) sub-module initially identifies the manipulation task that the robot needs to perform. The DS uses the information of the objects in the scene (output of the *environmental perception module*) as input. For the sake of simplicity, throughout the thesis, the objects involved in the manipulation task during demonstration, which are stored in the database, are referred to as *original objects,* and the objects present in the scene in the working phase are referred to as *current objects*.

For every manipulation task, a Q-Table is initialized and stored during the learning phase (section 3.3.1.2). The DS searches through the stored Q-Tables to find the manipulation task, in which the IDs of the current objects match the original objects in the stored actions. In case there are additional objects in the current scene in comparison to the original scene (scene during the demonstrations), these objects are considered obstacles. After the task identification, the DS suggests the sequence of actions using interactive Reinforcement Learning.

In Interactive Reinforcement Learning, the user is the external trainer who evaluates how good or bad the action suggestions from the RL agent are. Two main approaches in interactive RL are policy and reward shaping [137]. In policy shaping, the external trainer can replace the proposed action by the RL agent with a more suitable action, before it is executed. In reward shaping, the external trainer evaluates the performed action after the action is executed.

In the presented work, a policy shaping approach is selected. The user is able to select the action and prevent the robot from performing an action that is not preferable or even

safe. The reward function provides a positive reward to the RL agent every time the user confirms a suggested action, and a negative reward when the user rejects the suggested action. The reward function is shown by the equation (3.11):

$$R(g_\tau, a_\tau, g_{\tau+1}) = \begin{cases} 1 \; if \; user \; confirms/selects \; the \; suggested \; a_t \\ -5 \; if \; user \; rejects \; the \; suggested \; a_t \end{cases} \tag{3.11}$$

where g_τ is the current state in time τ, a_τ is the action in time τ, and $g_{\tau+1}$ is the new state in time τ+1. The absolute value of the negative reward is higher than the value of the positive reward, so the reinforcement learning can converge faster.

In the Reinforcement Learning algorithm, the Q-learning [138] method is selected. Q-learning is an off-policy method that learns the value of taking an action a_t from a given state g_t. The $Q(g_\tau, a_\tau)$ value is updated based on equation (3.12) [36]:

$$Q(g_\tau, a_\tau) \leftarrow (1 - \mathrm{a}) \cdot Q(g_\tau, a_\tau) + \mathrm{a} \cdot \left[R(g_\tau, a_\tau, g_{\tau+1}) + \delta \cdot \max_{a_{\tau+1} \in A} Q(g_{\tau+1}, a_{\tau+1}) \right] \tag{3.12}$$

where a $\in$ [0,1] is the learning rate and $\delta \in [0,1]$ is the discount factor. The discount factor determines how important the future rewards are. The parameters used in (3.12) are empirically set to a $= 0.3$ and $\delta = 0.3$.

Algorithm 1 shows the interactive Reinforcement Learning approach that is implemented in the presented work. In line 4 of algorithm 1, the policy on finding the action suggestion is modified based on the work presented in [139]:

$$a_\tau \leftarrow \mathrm{argmax}_{a \in A_g} Q(g_\tau, a) \tag{3.13}$$

where A_g is a subset of available actions in the current state g_τ. The algorithm suggests an action a_τ. If the Q-value of the suggested action is not positive, the robot asks the user to select an action (algorithm 1, lines 8 – 11). The selection of the action by the user helps the algorithm to converge faster. Otherwise the algorithm would have to exploit all possible actions and request feedback from the user for each action. An action is taken only if the reward has a positive value. It is not desirable to select an action that results in a reward that is zero or negative, as it may lead to an unwanted/dangerous state.

Algorithm 1: Decision support – Interactive Reinforcement Learning (Q-Learning)

Input: $Q(g, a)$
1. $u \leftarrow 1$ //Counter of selected actions.
2. **repeat**
3. **if** ($g_\tau \neq$ terminal)
4. $a_\tau \leftarrow \text{argmax}_{a \in A_g} Q(g_\tau, a)$ //Choose an a_τ with maximum $Q(g_\tau, a), a \in A_g$
5. **if** ($Q(g_\tau, a_\tau) > 0$ is true) & (user rejects a_τ) **then**
6. Update $Q(g_\tau, a_\tau)$ based on the equation (3.12)
7. **else**
8. **if** ($Q(g_\tau, a_\tau) > 0$ is not true) **then**
9. Ask user to select $a_\tau \in A$
10. $a_\tau \leftarrow$ selected by the user
11. **end if**
12. $\hat{a}_u \leftarrow$ selected a_τ
13. **if** (u=1) **then**
14. Robot Initialization
15. **else**
16. $\hat{\mu}_n \leftarrow$m-GMM // more details in Section 3.4.2.1.
17. $\hat{\beta} \leftarrow$GMR // more details in Section 3.4.2.2.
18. Take action $\hat{a}_u$ // including moving action.
19. Update $Q(g_\tau, a_\tau)$ based on the equation (3.12)
20. $g_\tau \leftarrow g_{\tau+1}$
21. $u \leftarrow u + 1$
22. **end if**
23. **end if**
24. **end if**
25. **if** ($g_\tau =$ terminal)
26. Ask user if $g_{\tau+1}$ is needed
27. **if** (user accepts $g_{\tau+1}$ is needed) **then**
28. $Q(g_{\tau+1}, a) \leftarrow 0$ // $\forall a \in A_g$
29. $g_\tau \leftarrow g_{\tau+1}$
30. **else**
31. exit program
32. **end if**
33. **end if**
34. **until** (user exits program)
35. **return** $Q(g, a)$

The $\hat{a}_u$ is a vector, in which the selected actions (confirmed by the user) are stored (algorithm 1, line 12). If the selected action is the first one (algorithm 1, lines 13 – 14), which means it will be either a home action or a start action, the robot confirms that its

position is where it is expected to be. In case the robot is not in the expected position, it will ask the user for assistance.

For the next actions selected by the user, the robot performs the selected action, and then the Q-value is updated based on the equation (3.12) and the algorithm moves forward to the new state (algorithm 1, lines 16 – 21). Line 18 of algorithm 1 means that the robot moves (moving action) and then performs the selected $\hat{a}_u^r$ action (grasp/release the object in case $\hat{a}_u^r$ is Close/Open, or do nothing after the moving action if $\hat{a}_u^r$ is Home/Start/End). For example, if the selected action is *Close –Cup info*, the robot has to move to the cup (moving action) and actuate (close) the gripper. The methodology of generating the moving actions (trajectories) is given in section 3.4.2. If the Q-value of the suggested action is positive and the user rejects the suggested action, the Q-value is updated and there is no change in the state of the robot (algorithm 1, lines 5 – 6).

Moreover, when the state is terminal, the algorithm asks the user to confirm that no more states are needed (algorithm 1, lines 25 – 26). In case the user confirms that additional state is needed, the algorithm will add an additional row to the Q-Table with zero awards (algorithm 1, lines 27 – 29) and the whole process is repeated until the user confirms final state or exits the program (algorithm 1, lines 30 – 34). At the end, algorithm 1 stores the updated Q-Table $Q(g, a)$ (algorithm 1, line 35).

3.4.2 Adaptation at Trajectory Level (Low-level)

This section presents the methods to generate the necessary moving actions for the learned task. A novel GMM modification algorithm is used for the adaptation at trajectory level. The GMM modification (m-GMM) algorithm is first presented by Kyrarini et al. in [1] and then extended by Kyrarini et al. in [4]. The m-GMM modifies the mean values of the learned Gaussian Mixture Model for the moving actions (sub-trajectories) in order to adapt to the new pose of the objects and to avoid collision with obstacles. The output of the m-GMM is used by the Gaussian Mixture Regression (GMR) to generate the moving action (low-level). The generated trajectory (moving action) by the GMR is connected directly to the modified GMM. The m-GMM is a very crucial part of the presented framework.

3.4.2.1 Modification of the learned Gaussian Mixture Model

In this section, the m-GMM algorithm for the modification of the mean values of learned Gaussians is explained in order to successfully adapt the moving actions of the task to the new environmental conditions. The new environmental conditions arise due to changes in pose of the objects or the persons, or due to obstacles. The objects are positioned on a working table (workspace) and rotations around the vertical axis are only considered. The notations for section 3.4.2.1 are summarized in Table 8.

Table 8: Notations for section 3.4.2.1.

Symbol	Explanation
N	Total number of Gaussians in the GMM model
n	$n = \{1, \dots, N\}$ the number of the Gaussian respectively
dim	$dim = \{1, \dots, 7\}$ represents the dimensions *{x, y, z, qx, qy, qz, qw}* respectively
μ_n	Mean values of the learned GMM for the 7 dimensions of the robot's end-effector
$\mu_n(dim)$	Mean values of the learned GMM for the dim dimensions of the robot's end-effector
$\hat{\mu}_n$	Mean values of the modified GMM for the 7 dimensions of the robot's end-effector (output of m-GMM)
u	Counter of selected actions
$\hat{a}_u$	Selected action $\hat{a}_u = \{\hat{a}_u^r, \hat{a}_u^{obj}\}$
$\hat{a}_u^r$	Robot actions: {*Start, End, Close, Open, Home*}
$\hat{a}_u^{obj}$	Information (ID, size, pose) of the related object to the selected action. $obj = \{orig\_obj, cur\_obj\}$ for the original and current object respectively
$obst$	Information (ID, size, pose) of the obstacles – non related objects for the selected action
dim_obj	$dim\_obj = \{1, \dots, 11\}$ represents the information of the object *{ID , length, width, height, x, y, z, qx, qy, qz, qw}*, respectively
$\hat{a}_u^{obj}(dim\_obj)$	Information dim_obj of the related object to the selected action. $obj = \{orig\_obj, cur\_obj\}$
$obst(dim\_obj)$	Information dim_obj of the obstacles – non related objects for the selected action

The m-GMM has the advantage that it does not require demonstrations with different environmental conditions. The m-GMM takes into consideration the constraints of the demonstrated trajectories which are learned by the GMM. The inputs of the m-GMM are:

- the learned GMM (μ_d) of the sub-trajectories between the actions $\hat{a}_{u-1}$ and $\hat{a}_u$,
- the selected actions $\hat{a}_{u-1}$, $\hat{a}_u$ which include:
    - the robot actions $\hat{a}^r_{u-1}$ and $\hat{a}^r_u$ (robot actions: {*Start, End, Close, Open, Home*}),
    - the information (ID, size, pose) of the related original objects $\hat{a}^{orig\_obj}_{u-1}$, $\hat{a}^{orig\_obj}_u$ and of the related current objects $\hat{a}^{cur\_obj}_{u-1}$, $\hat{a}^{cur\_obj}_u$ to the actions $\hat{a}_{u-1}$, $\hat{a}_u$, respectively, and
- the information (ID, size, pose) of the non-related objects (obstacles) to the selected actions, denoted as *obst*.

Algorithm 2 explains how the modification of Gaussian Means (m-GMM) is performed. It is generic and it does not depend on the number of Gaussians *N* (as long as the Gaussians are more than 3). Algorithm 2 is applied to each of the seven-dimensional spatial variables separately (algorithm 2, line 12) and consists of the following steps, with the goal of modifying the Gaussians for a specific environment.

➢ *<u>Step 1:</u> Modify the mean values of the important Gaussians (algorithm 2, lines 3 – 27)*

The algorithm modifies the mean values of the important Gaussians of the learned GMM. The important Gaussians are the first and the last Gaussians of the GMM, which represent the data-points close to the location of picking up or placing an object.

If the robot action $\hat{a}^r_{u-1}$ or $\hat{a}^r_u$ is equal to 'home', then the new mean value of the first $\hat{\mu}_1$ or the last $\hat{\mu}_N$ Gaussian is equal to the mean value of the first μ_1 or the last μ_N learned Gaussian, respectively (algorithm 2, lines 14 – 15 or lines 21 – 22, respectively). Otherwise, the new mean values of the important Gaussians are modified based on the new position and orientation of the related current objects. The new mean values of the important Gaussians for the dimensions *{x,y,z}* are modified as shown in lines 16 – 18 and 23 – 25. The new mean values of the important Gaussians for the orientation (quaternion)

are modified as shown in lines 2 – 11. In particular, the lines 5 and 10 calculate the product of two given quaternions. An example of the modification of the important Gaussians is shown in Figure 17b for a learned GMM (Figure 17a).

Algorithm 2: m-GMM; Modification of Gaussian Means of the GMM

Input: $\mu_n, u, \hat{a}_{u-1}, \hat{a}_u, obst$
Output: $\hat{\mu}_n$
// the quaternion values are calculated for the first and last Gaussian. The *dim=4,5,6,7*
// represents the quaternion for the mean values of the GMM and *dim_obj=8,9,10,11 the*
// *quaternion for the objects.*

1. **if** $(u > 1)$ **then**
2. **if** $(\hat{a}_{u-1}^{r} \neq Home)$ **then**
3. $qx_{u-1} \leftarrow \left[\hat{a}_{u-1}^{cur\_obj}(8) - \hat{a}_{u-1}^{orig\_obj}(8)\right], qy_{u-1} \leftarrow \left[\hat{a}_{u-1}^{cur\_obj}(9) - \hat{a}_{u-1}^{orig\_obj}(9)\right],$
4. $qz_{u-1} \leftarrow \left[\hat{a}_{u-1}^{cur\_obj}(10) - \hat{a}_{u-1}^{orig\_obj}(10)\right], qw_{u-1} \leftarrow \left[\hat{a}_{u-1}^{cur\_obj}(11) - \hat{a}_{u-1}^{orig\_obj}(11)\right]$
5. $[\hat{\mu}_1(4), \hat{\mu}_1(5), \hat{\mu}_1(6), \hat{\mu}_1(7)] \leftarrow [qx_{u-1}, qy_{u-1}, qz_{u-1}, qw_{u-1}] * [\mu_1(4), \mu_1(5), \mu_1(6), \mu_1(7)]$
6. **endif**
7. **if** $(\hat{a}_{u}^{r} \neq Home)$ **then**
8. $qx_{u} \leftarrow \left[\hat{a}_{u}^{cur\_obj}(8) - \hat{a}_{u}^{orig\_obj}(8)\right], qy_{u} \leftarrow \left[\hat{a}_{u}^{cur\_obj}(9) - \hat{a}_{u}^{orig\_obj}(9)\right],$
9. $qz_{u} \leftarrow \left[\hat{a}_{u}^{cur\_obj}(10) - \hat{a}_{u}^{orig\_obj}(10)\right], qw_{u} \leftarrow \left[\hat{a}_{u}^{cur\_obj}(11) - \hat{a}_{u}^{orig\_obj}(11)\right]$
10. $[\hat{\mu}_N(4), \hat{\mu}_N(5), \hat{\mu}_N(6), \hat{\mu}_N(7)] \leftarrow [qx_{u}, qy_{u}, qz_{u}, qw_{u}] * [\mu_N(4), \mu_N(5), \mu_N(6), \mu_N(7)]$
11. **endif**
12. **for**$(dim = 1; dim < 8, dim++)$
13. $dim\_obj = dim + 4$ // the first 4 *dim_obj* are ID, width, length, height
14. **if** $(\hat{a}_{u-1}^{r} = Home)$ **then**
15. $\hat{\mu}_1(dim) \leftarrow \mu_1(dim)$
16. **else** // $\hat{a}_{u-1}^{r}$ is Start or End or Open or Close
17. **if** $(dim < 4)$
18. $\hat{\mu}_1(dim) \leftarrow \mu_1(dim) + \left[\hat{a}_{u-1}^{cur\_obj}(dim\_obj) - \hat{a}_{u-1}^{orig\_obj}(dim\_obj)\right]$
19. **endif**
20. **endif**
21. **if** $(\hat{a}_{u}^{r} = Home)$
22. $\hat{\mu}_N(dim) \leftarrow \mu_N(dim)$
23. **else** // $\hat{a}_{u}^{r}$ is Start or End or Open or Close
24. **if** $(dim < 4)$
25. $\hat{\mu}_N(dim) \leftarrow \mu_N(dim) + \left[\hat{a}_{u}^{cur\_obj}(dim\_obj) - \hat{a}_{u}^{orig\_obj}(dim\_obj)\right]$
26. **endif**
27. **endif**
28. $\hat{\mu}_n(dim) \leftarrow modify\_GMM[\mu_n(dim), \hat{\mu}_1(dim), \hat{\mu}_N(dim), dim]$
29. **endfor**
30. $\hat{\mu}_n \leftarrow modify\_obstGMM[\hat{\mu}_n, obst, \hat{a}_{u-1}, \hat{a}_u]$
31. **endif**
32. **return** $\hat{\mu}_n$

Product of two given quaternions

For two given quaternions $q_1 = \{qx_1, qy_1, qz_1, qw_1\}$ and $q_2 = \{qx_2, qy_2, qz_2, qw_2\}$, the quaternion product $q = \{qx, qy, qz, qw\}$ is calculated as follows:

$$qx = qw_1 \cdot qx_2 + qx_1 \cdot qw_2 - qy_1 \cdot qz_2 + qz_1 \cdot qy_2$$

$$qy = qw_1 \cdot qy_2 + qx_1 \cdot qz_2 + qy_1 \cdot qw_2 - qz_1 \cdot qx_2$$

$$qz = qw_1 \cdot qz_2 - qx_1 \cdot qy_2 + qy_1 \cdot qx_2 + qz_1 \cdot qw_2$$

$$qw = qw_1 \cdot qw_2 - qx_1 \cdot qx_2 - qy_1 \cdot qy_2 - qz_1 \cdot qz_2$$

- *Step 2: Modify the mean values of the remaining Gaussians (algorithm 2, line 28)*

The function $modify\_GMM$ is used to modify the remaining Gaussians and it is presented in Algorithm 3. First, the function identifies and modifies the Gaussians that are close to the first and last (important) Gaussians. The Gaussians close to the first one are identified by calculating the absolute difference between the mean values of each Gaussian and the first Gaussian in the learned GMM. If the absolute difference between the Gaussians is smaller than a predefined threshold value, then the Gaussian is considered to be a close Gaussian and is modified (algorithm 3, lines 5 – 13). A similar procedure is performed for the Gaussians close to the last Gaussian (algorithm 3, lines 14 – 22). The threshold values (algorithm 3, lines 1 – 4) are selected empirically. The modification of the close Gaussians enables the generation of an accurate trajectory (by using GMR) for object manipulation. An example of the modification of the close Gaussians is shown in Figure 17c. Lastly, algorithm 3 (lines 23 – 29) modifies the intermediate Gaussians (Gaussians between the close Gaussians) to provide a smoother generated trajectory by the GMR. An example of the modification of the intermediate Gaussians is shown in Figure 17d.

Algorithm 3: Function for modification of remaining Gaussian means (close and intermediate Gaussians)

function $modify\_GMM[\mu_n(dim), \hat{\mu}_1(dim), \hat{\mu}_N(dim), dim]$
//$\mu_n(dim)$ is a vector which consists of all the GMM values for the dimension *dim*
//$\hat{\mu}_1(dim), \hat{\mu}_N(dim)$ are the first and last value of the modified GMM for the
//dimension *dim*

```
if (dim < 4) then
    c_μ ← 0.02 //x, y, z dimension: 2 cm
else
    c_μ ← 0.015 //qx, qy, qz, qw dimension: 0.015
//Find and modify close Gaussians to the first Gaussian
for(n = 2; n ≤ N, n++) do
    if (|μ_n(dim) − μ_1(dim)| < c_μ) then
        μ̂_n(dim) ← μ_n(dim) + [μ̂_1(dim) − μ_1(dim)]
    else
        m ← n
        break
    endif
endfor
//Find and modify close Gaussians to the last Gaussian
for(n = N − 1; n > 1, n − −) do
    if (|μ_n(dim) − μ_N(dim)| < c_μ) then
        μ̂_n(dim) ← μ_n(dim) + [μ̂_N(dim) − μ_N(dim)]
    else
        l ← n
        break
    endif
endfor
//Find and modify intermediate Gaussians
if (m ≤ l) then
    diff ← [(μ̂_{m−1}(dim) − μ_{m−1}(dim)) + (μ̂_{l+1}(dim) − μ_{l+1}(dim))]/2
    for(n = m; n ≤ l, n++) do
        μ̂_n(dim) ← μ_n(dim) + diff
    endfor
endif
return μ̂_n(dim)
```

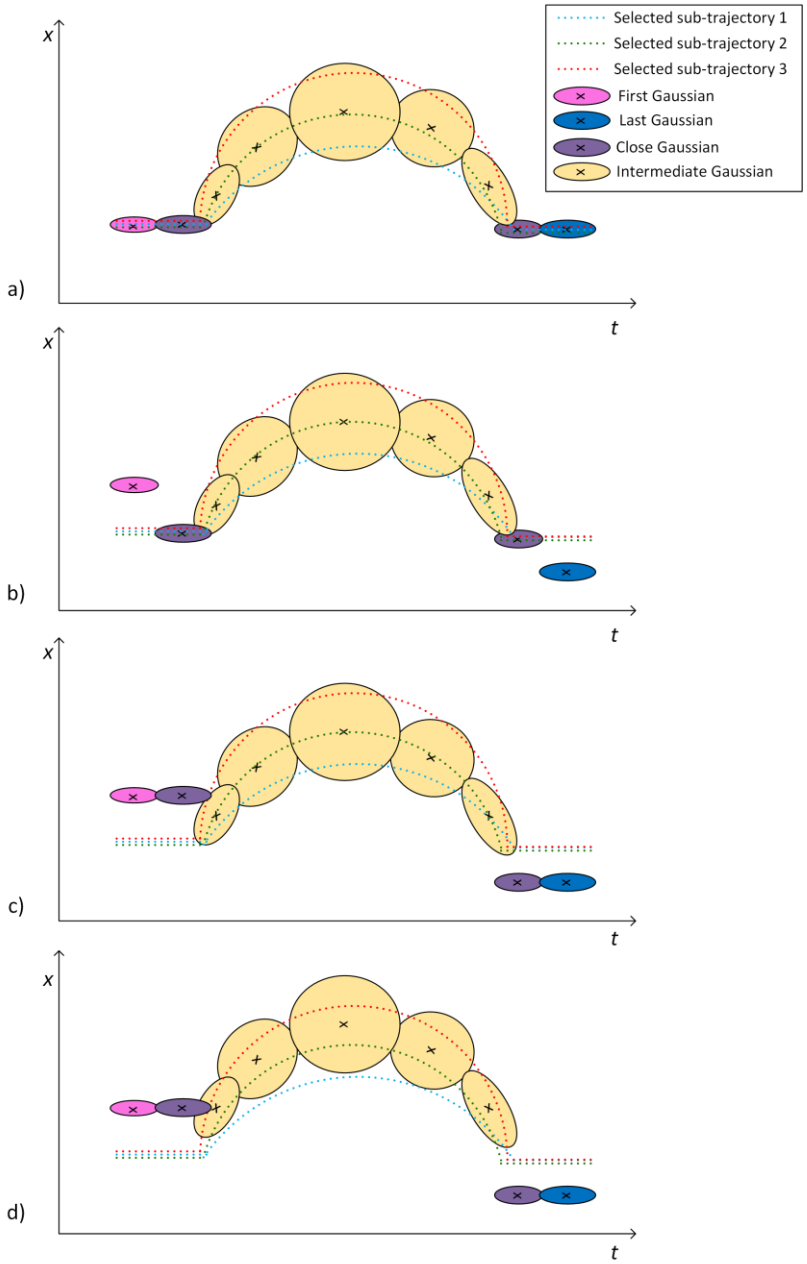


Figure 17: Explanatory illustration of the m-GMM algorithm a) the learned GMM and the identification of the Gaussians, b) the modification of the first and last Gaussians, c) the modification of the close Gaussians, d) the modification of the remaining Gaussians.

- *<u>Step 3:</u> Find and Modify Gaussians that are in collision with obstacles (algorithm 2, line 30)*

The function $modify\_obstGMM$ determines which Gaussians are in collision with the identified obstacles. If a mean value of the $\hat{\mu}_n$ is inside the bounding box of an object, then this Gaussian is flagged as 'Gaussian in obstacle'. The bounding box of the involved

objects is oversized by a factor (in the presented framework the oversized factor is 120%) to ensure safe avoidance. Furthermore, if the $\hat{a}_{u-1}^{r} = Close$ (which means the $\hat{a}_{u-1}^{cur\_obj}$ object is grasped by the robot), the dimensions of the grasped object $\hat{a}_{u-1}^{cur\_obj}$ are also taken into account to determine the possible collision with obstacles. One example of the Gaussians in obstacle is shown in Figure 18.

After the Gaussians in obstacle are identified, the function modifies the Gaussians to avoid collision with obstacles. The mean values of Gaussians in obstacles are modified by adding the height of the obstacles to the z dimension of the $\hat{\mu}_n$ plus a safety distance. If the $\hat{a}_{u-1}^{r} = Close$, then the mean values of Gaussians in obstacles are modified by adding the height of the obstacle plus the height of the grasped current object to the z-dimension of the $\hat{\mu}_n$ plus a safety distance. The obstacle avoidance happens in z-dimension, which is the vertical axis. The assumption is that the position of obstacles does not prevent the completion of the task, i.e. an obstacle is not close to the target position.

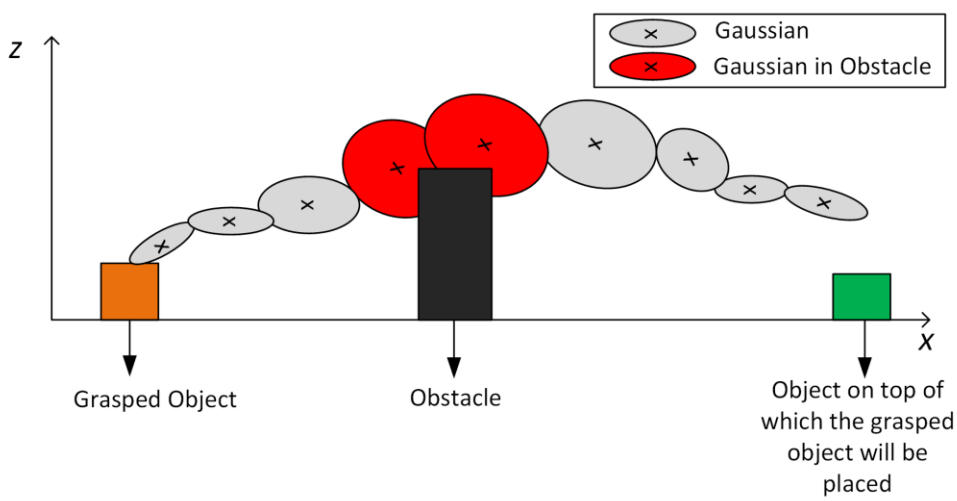


Figure 18: Explanatory illustration of the 'Gaussians in Obstacle' in 2D.

3.4.2.2 Gaussian Mixture Regression (GMR)

Gaussian Mixture Regression (GMR) is a method which is used to produce the trajectory of the end-effector and it can be used to control the robot efficiently [130]. As input, GMR uses the parameters of the GMM and generates a fast and optimal output [130].

The output of the m-GMM ($\hat{\mu}_n$) is used by the GMR to generate the trajectory $\hat{\beta}$ that is adapted to the changed environments. The output $\hat{\beta}$ is calculated by the equation (3.14) [130]:

$$\hat{\beta} = \left\{ \beta_t, \sum_{n=1}^{N} \zeta_n \hat{\beta}'_{n,s} \right\} \tag{3.14}$$

where: $\zeta_n = p(\beta_t|n) / \sum_{n=1}^{N} p(\beta_t|n)$ and $\hat{\beta}'_{n,s} = \hat{\mu}_{n,s} + \Sigma_{n,st} \left(\Sigma_{n,t}\right)^{-1} \left(\beta_t - \mu_{n,t}\right)$, $\forall n = \{1, .., N\}$. The trajectory generated by the GMR is given to the robot controller so the robot can reproduce it. The velocity at which the robot reproduces the trajectory is predefined. Figure 19 illustrates the generated trajectory by the GMR based on the given modified GMM (output of m-GMM).

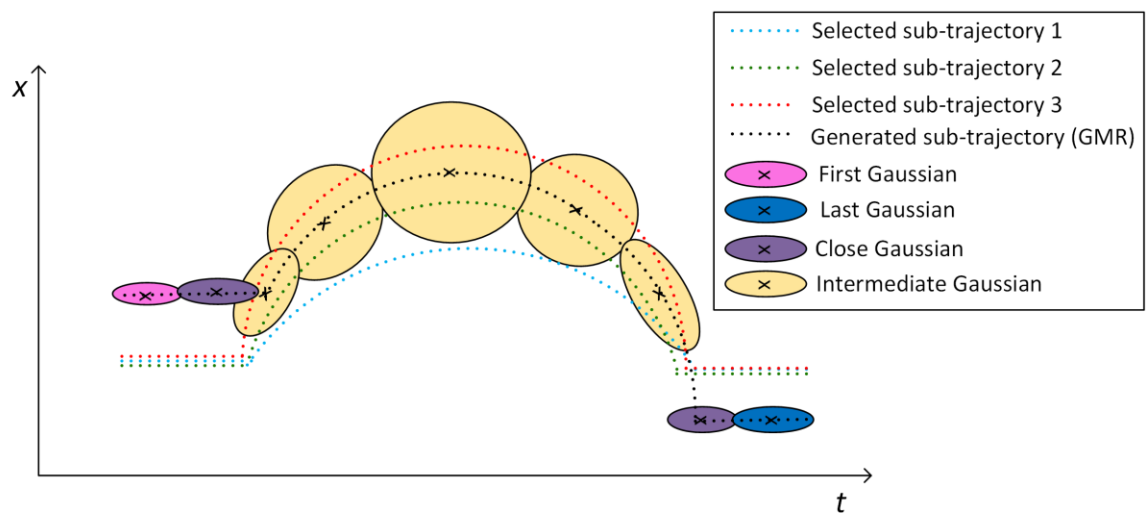


Figure 19: Explanatory illustration of the generated sub-trajectory by the GMR.

3.4.3 Robot Learning User's Preferences

The *Robot Learning User's Preferences module* is responsible for learning the preferable sequence of actions for a particular user. This is important when a robot cooperates with several users to perform a task. For example, in an assembly line, a robot may have to collaborate with three different users during the day. Some users may prefer a different sequence of actions to complete the assembly task than others.

Reinforcement Learning is used for personalized learning of preferences [140] [141] [142] [143]. A default RL agent is initially built and then adapted to each particular user [143]. Each user provides rewards to express satisfaction towards the suggested actions by the RL agent.

In the presented framework, the RL agent (Q-table) is initialized by the human demonstrations (section 3.3.1.2) and it is assigned as default Q-table. During the working phase, it is updated by the *DS* sub-module (section 3.4.1). For each particular user, a different Q-table is updated during the working phase. Algorithm 4 shows the method that enables the robot to learn the preferable sequence of actions for each user. The algorithm initially requests the user's name (algorithm 4, line 1). Each user has a unique username. Subsequently, the algorithm searches throughout the user database to retrieve the user's Q-table (algorithm 4, lines 2 – 3). If the user cooperates with the robot for the first time, their username is added to the database and the default Q-table is used (algorithm 4, lines 4 – 6). The user's Q-table is used as input to algorithm 1 (section 3.4.1) and the updated Q-table that is output of algorithm 1 is stored as the updated user's Q-table in the robot database (algorithm 4, line 8).

Algorithm 4: Robot Learning User's Preferences

1. $user \leftarrow$ request username
2. **if** $(user \in Data\_user)$ // $Data\_user$ is the user database
3. $\quad Q(g,a) \leftarrow Q_{user}(g,a)$
4. **else**
5. $\quad$ add $user$ in $Data\_user$
6. $\quad Q(g,a) \leftarrow Q_{default}(g,a)$
7. **end if**
8. $Q_{user}(g,a) \leftarrow$ Algorithm 1

4. Robot Learning of Industrial Assembly Tasks from Multiple Human Demonstrations – Application in Industrial Robotics

In this chapter, the presented robot learning from multiple human demonstrations is evaluated for two industrial assembly tasks. The users were able to teach a dual-arm industrial robot assembly tasks. After the demonstrations and the learning, the robot performed the tasks, even when the positions of the objects have changed (in comparison to the demonstration). To demonstrate the feasibility of the RLfD framework, two assembly tasks were selected and three small studies were conducted. The experimental results are explained in detail as follows.

4.1 Industrial Robotic Platform

The dual-arm industrial *pi4 Workerbot 3* [144] was selected as a collaborative robotic platform. It consists of two 6-Degrees of Freedom (DoF) UR10 robotic arms [145], which support kinesthetic teaching. As illustrated in Figure 20a, each robotic arm of the Workerbot3 is equipped with an industrial vacuum gripper. Furthermore, a two-finger gripper is mounted on the left robotic arm. All three grippers have two actuation states; “On” and “Off” denoting the actuated and not-actuated gripper state. The two-finger

gripper and the vacuum gripper mounted on the left arm cannot be used simultaneously. In Figure 20b, the world coordinate system is presented, which is located at the robot's base.

The workplace of the robot, which is shown in Figure 20a, is a table located in front of the Workerbot. A *Kinect for Xbox One* camera [146], which was produced by Microsoft, is mounted onto the head of the Workerbot and it is used as the vision-sensor, which provides information about the workplace. The Kinect contains a color camera (RGB) and a depth camera system, comprising an Infrared (IR) camera and an IR projector.

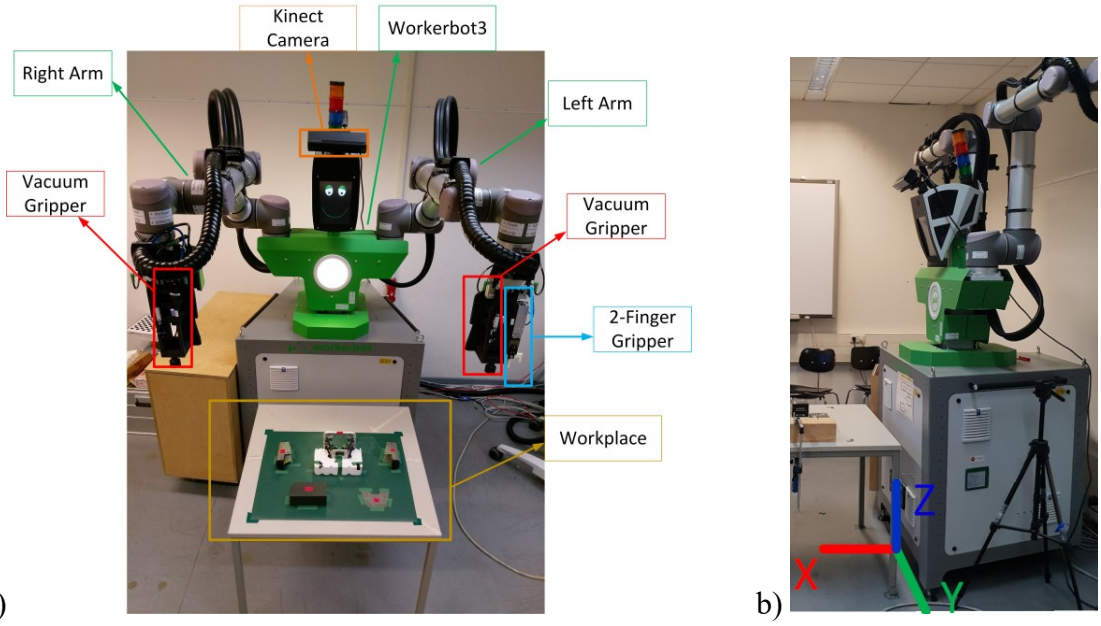


Figure 20: a) The collaborative robot Workerbot3 in the 'Home' pose, b) The world coordinate system located at the robot's base.

4.2 Robot Gripper Assembly Task (Task 1)

For the evaluation of the presented framework, an assembly of a robot gripper is selected as an industrial human-robot collaborative task and has been presented by Kyrarini et al. in [1]. The assembly of the robot gripper consisting of 5 parts is illustrated in Figure 21. As shown, there are four gripper's parts (black part, top part, left and right side part), which shall be assembled with the fifth part (base part), to complete the assembly task. Several users, firstly, demonstrated the task and the robot learned the task with the help of the presented RLfD framework. In working phase, the robot has the role

to manipulate the gripper parts so to place them in appropriate places next to each other, while the user (human collaborator) screws the parts together to successfully complete the assembly task.

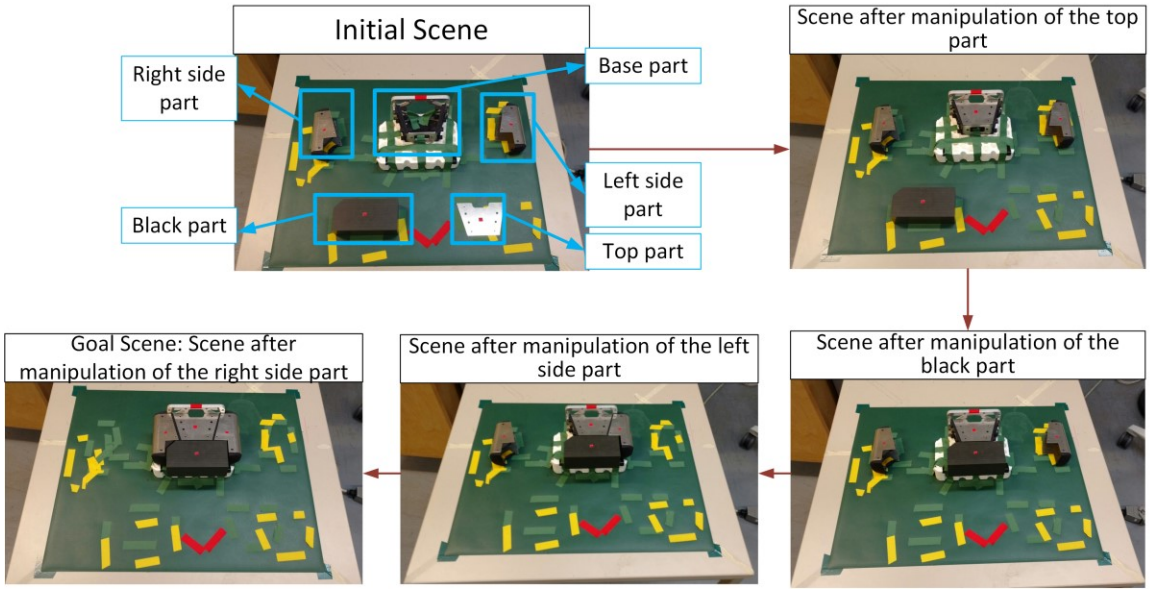


Figure 21: Robot Gripper Assembly Task (Task 1). (adapted from [1])

4.2.1 Human Demonstrations of the Robot Gripper Assembly Task

Three participants were asked to provide demonstrations of the robot gripper assembly task (task 1), shown in Figure 21. The method that was selected to provide demonstrations is kinesthetic teaching (section 3.2.1.1), as shown in Figure 22. Out of 3 participants, 1 is female and 2 are males. The average age was 29.6±3.2 years old. All participants gave their informed, signed consent for the participation. All participants had ten minutes to get familiar with the kinesthetic teaching approach, before they were asked to demonstrate the task.

Each demonstrator was instructed to follow the sequence of actions, shown in Figure 21. The manipulation of each four gripper's parts (top part, black part, left and right side part) is performed through the sequence of actions:

- pick the gripper's part,
- place it at the base part as shown in Figure 21, and
- move the robotic arm away from the working table.

The complete task was demonstrated three times by each demonstrator in the following order:

1. manipulation of the top part with the left robotic arm,
2. manipulation of the black part with the right robotic arm,
3. manipulation of the left side part with the left robotic arm, and
4. manipulation of the right side part with the right robotic arm.

During the demonstrations, both robotic arms were used, but never at the same time. The reason was that the demonstrator used both of their hands to guide one robotic arm via kinesthetic teaching. Therefore, it was not physically possible to guide both robotic arms at the same time.

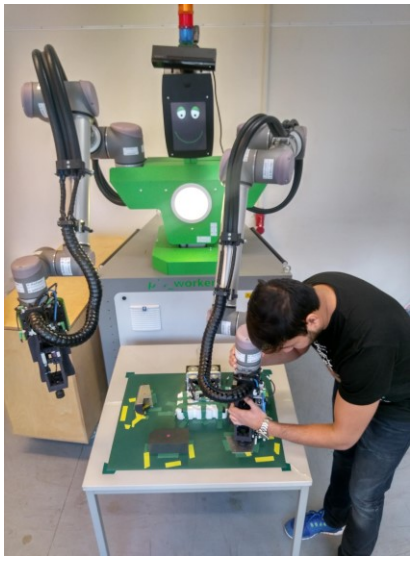


Figure 22: A user demonstrates the robot gripper assembly task using kinesthetic teaching.

4.2.2 Robot Learning and Working Phase of Robot Gripper Assembly Task

The next step, after the users demonstrated the robot gripper assembly task, was the robot learning phase. In total, nine demonstrations (three participants providing three demonstrations) were used as input to the robot learning phase.

- **Robot Learning Phase (Offline)**

The nine demonstrations were first processed by the Symbolic Task Learning (High-level) module, presented in section 3.3.1. The ATSSA module was used to generate the sequence of actions. However, since the robotic platform has two arms and three grippers, the robot actions were updated to include information about which arm and which gripper is used. The robot action a^r was updated to include the identification of the gripper as follows: $a^r \in$ {*LV Start, L2 Start, RV Start, LV End, L2 End, RV End, LV Close, L2 Close, RV Close, LV Open, L2 Open, RV Open, LV Home, L2 Home, RV Home*}, where *LV, L2, RV* denote vacuum gripper of the left arm, 2-finger gripper of the left arm and vacuum gripper of the right arm, respectively. The sequence of actions generated by the ATSSA sub-module for the demonstrations was:

1. *LV Home – Home*,
2. *LV Close – Top part info*,
3. *LV Open – Base part info*,
4. *LV End – Table info*,
5. *RV Home – Home*,
6. *RV Close – Black part info*,
7. *RV Open – Base part info*,
8. *RV End – Table info*,
9. *LV Start – Table info,*
10. *LV Close – Left side part info*,
11. *LV Open – Base part info*,
12. *LV End – Table info*,
13. *RV Start – Table info,*
14. *RV Close – Right side part info*,
15. *RV Open – Base part info*,
16. *RV End – Table info*.

The poses of the gripper's parts and the table during the demonstrations were estimated with respect to the world coordinate system by the *environmental perception module v1* and the results are shown in Table 9. Based on equations (3.1) and (3.2), and the output of the ATSSA sub-module, the initialization of the Q-table for the demonstrated robot gripper assembly task is shown in Table 10. As it is observed, all the demonstrations have followed the same sequence of actions for completing the robot assembly task.

Table 9: Estimated pose of objects by the *environmental perception module v1* during the demonstrations of the robot gripper assembly task.

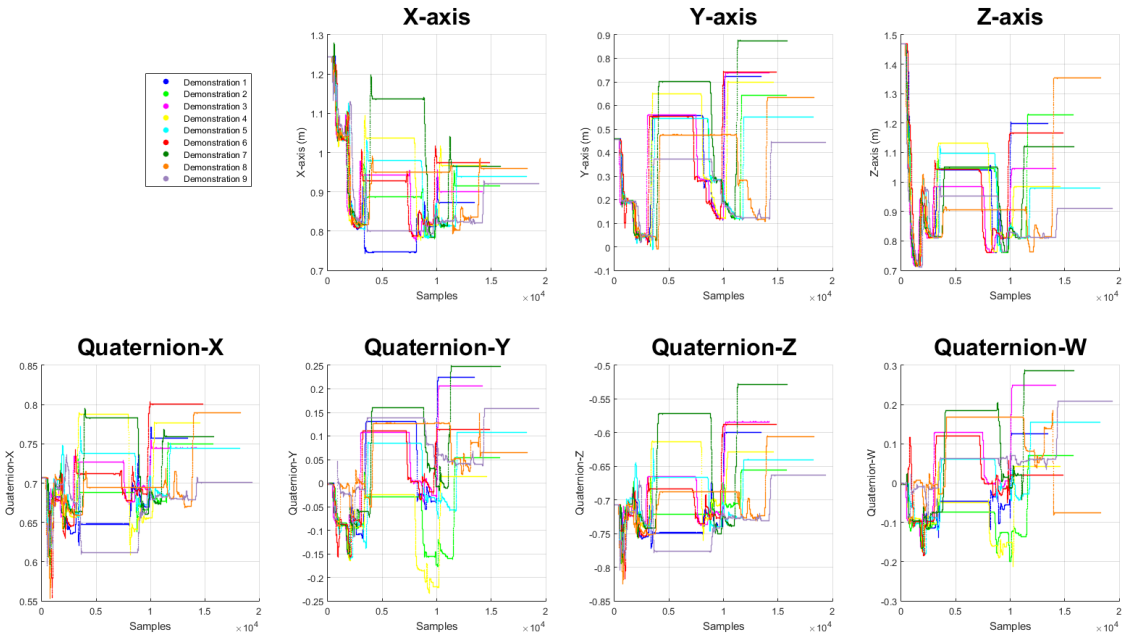
Object	**Initial Scene – Position and Orientation**			
	X (m)	**Y (m)**	**Z (m)**	**Rotation around Z-axis (°)**
Top part	1.043	0.194	0.729	0.201
Left side part	0.809	0.279	0.736	89.930
Black part	1.041	-0.076	0.755	0.202
Right side part	0.818	-0.196	0.736	90.200
Base part	0.819	0.046	0.755	0.207
Table	0.812	0.026	0.720	0.004

Table 10: Initialization of the Q-Table for the demonstrated robot gripper assembly task.

	Actions *a*											
States *g*	*LV Home – Home*	*LV Close – Top part info*	*LV Open – Base part info*	*LV End – Table info*	*RV Home – Home*	*RV Close – Black part info*	*RV Open – Base part info*	*RV End – Table info*	*LV Start – Table info*	*LV Close – Left side part info*	*RV Start – Table info*	*RV Close – Right side part info*
1	**1**	0	0	0	0	0	0	0	0	0	0	0
2	0	**1**	0	0	0	0	0	0	0	0	0	0
3	0	0	**1**	0	0	0	0	0	0	0	0	0
4	0	0	0	**1**	0	0	0	0	0	0	0	0
5	0	0	0	0	**1**	0	0	0	0	0	0	0
6	0	0	0	0	0	**1**	0	0	0	0	0	0
7	0	0	0	0	0	0	**1**	0	0	0	0	0
8	0	0	0	0	0	0	0	**1**	0	0	0	0
9	0	0	0	0	0	0	0	0	**1**	0	0	0
10	0	0	0	0	0	0	0	0	0	**1**	0	0
11	0	0	**1**	0	0	0	0	0	0	0	0	0
12	0	0	0	**1**	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	**1**	0
14	0	0	0	0	0	0	0	0	0	0	0	**1**
15	0	0	0	0	0	0	**1**	0	0	0	0	0
16	0	0	0	0	0	0	0	**1**	0	0	0	0

Subsequently, the nine demonstrations were processed by the Skill Learning at Trajectory Level (Low-level) module, presented in section 3.3.2. Figure 23 shows the nine demonstrated trajectories for the end-effector of the left and right robotic arm. As it can be seen, there is a variety between the demonstrations.

a) Demonstrated end-effector's trajectories of the left robotic arm



b) Demonstrated end-effector's trajectories of the right robotic arm

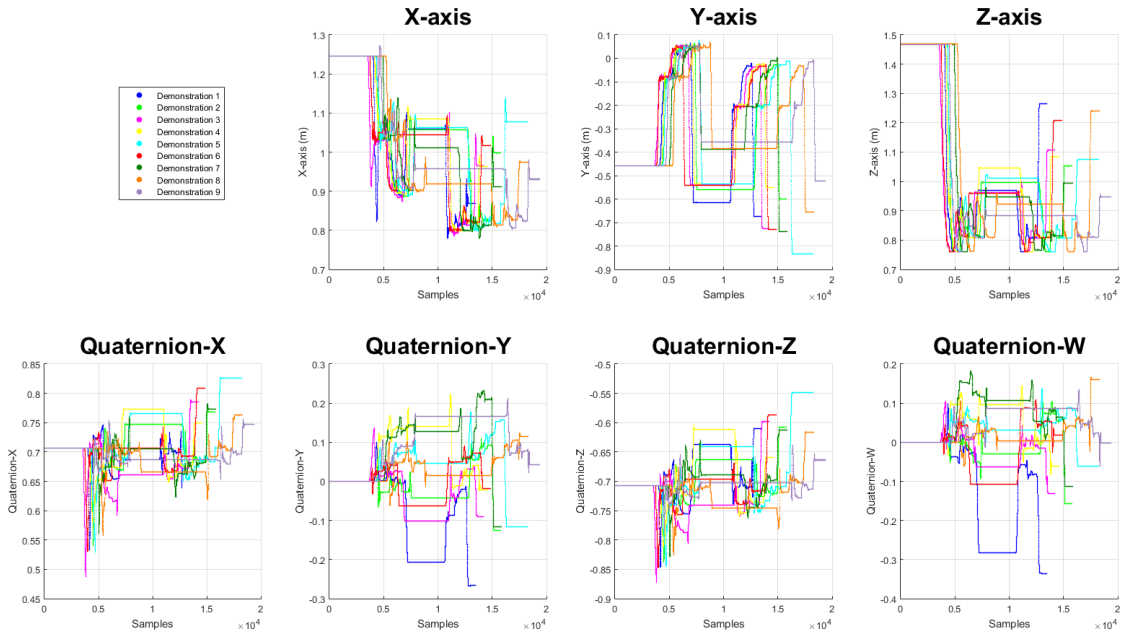


Figure 23: Demonstrated end-effector's trajectories of the robot gripper assembly task using the a) left and b) right robotic arm.

The demonstrated trajectories of the left robotic arm split into $G_L - 1$ sub-trajectories, where G_L is the total number of states for the left arm. Similarly, the demonstrated trajectories of the right robotic arm split into $G_R - 1$ sub-trajectories, where G_R is the total number of states for the right arm. Figure 24 presents one of the demonstrations for the robot gripper assembly task. The same procedure was performed for all the demonstrations. For each robotic arm, the trajectory was split into 7 sub-trajectories. The 4th sub-trajectory for each robotic arm was not used for further processing as the arm stays steady during that time (Figure 24).

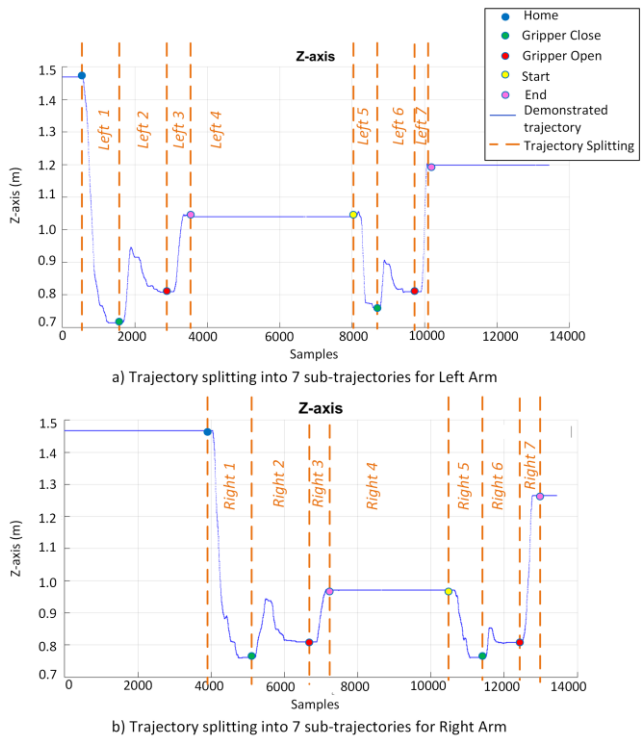


Figure 24: Splitting of trajectory for the demonstrated robot gripper assembly task. A) Splitting of trajectory into 7 sub-trajectories for left arm, b) Splitting of trajectory into 7 sub-trajectories for right arm. (adapted from [1])

The next step was to group the sub-trajectories, apply the Ramer-Douglas-Peucker simplification method, and select the similar sub-trajectories in a group, as explained in section 3.4.2.1. The threshold value c in equation (3.5) was selected as 30. However, if no similar sub-trajectory was found, the threshold value increased by 10, until at least one similar demonstration was found. Table 11 presents the groups and the selected similar demonstrations for each group of sub-trajectories. Additionally, Table 11 shows the

association between the sub-trajectories and the high-level actions. The simplified sub-trajectories of the groups Right 1, 2, and 3 for the manipulation of the black part using the right arm are shown in Figure 25, Figure 26, and Figure 27, respectively. The simplified sub-trajectories for the manipulation of the other gripper's parts are shown in Appendix C.I.

Table 11: Groups of sub-trajectories and selection of similar sub-trajectories for the robot gripper assembly task.

Group	**Sub-trajectories between the high-level actions**	**Reference sub-trajectory (demonstration number)**	**Similar sub-trajectories (demonstration number)**	**Threshold value *c* from equation (3.5)**
Left 1	*LV Home – Home and LV Close – Top part info*	7	1, 3, 4	30
Left 2	*LV Close – Top part info and LV Open – Base part info*	4	5, 6	30
Left 3	*LV Open – Base part info and LV End – Table info*	6	3	30
Right 1	*RV Home – Home and RV Close – Black part info*	6	3, 8	40
Right 2	*RV Close – Black part info and RV Open – Base part info*	8	5, 6, 9	40
Right 3	*RV Open – Base part info and RV End – Table info*	6	3	40
Left 5	*LV Home – Home and LV Close – Left side part info*	3	5	30
Left 6	*LV Close – Left side part info and LV Open – Base part info*	6	3, 5	30
Left 7	*LV Open – Base part info and LV End – Table info*	6	1, 8	50
Right 5	*RV Home – Home and RV Close – Right side part info*	5	2	30
Right 6	*RV Close – Right side part info and RV Open – Base part info*	9	2, 5	30
Right 7	*RV Open – Base part info and RV End – Table info*	4	3, 9	40

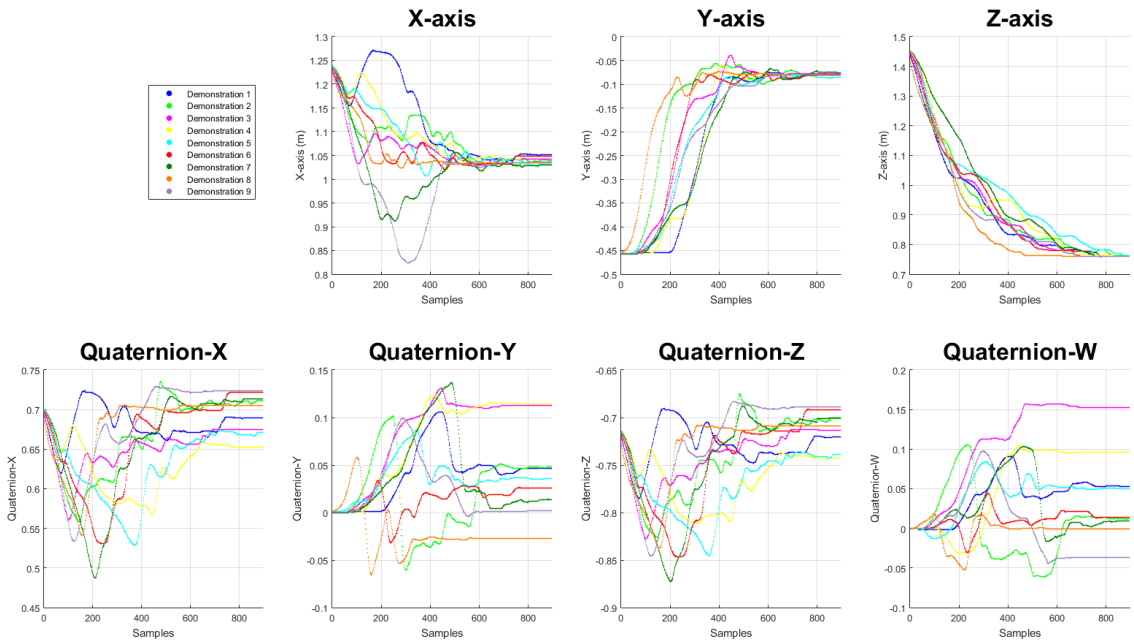


Figure 25: The sub-trajectories of the Right 1 group after Ramer-Douglas-Peucker simplification – Robot gripper assembly task.

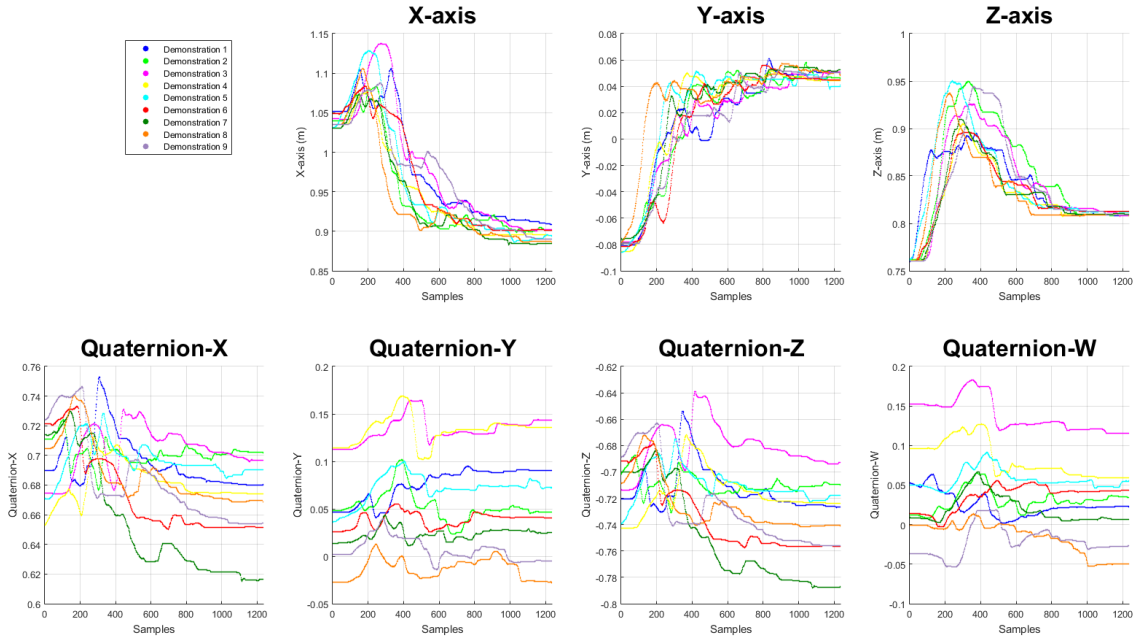


Figure 26: The sub-trajectories of the Right 2 group after Ramer-Douglas-Peucker simplification – Robot gripper assembly task.

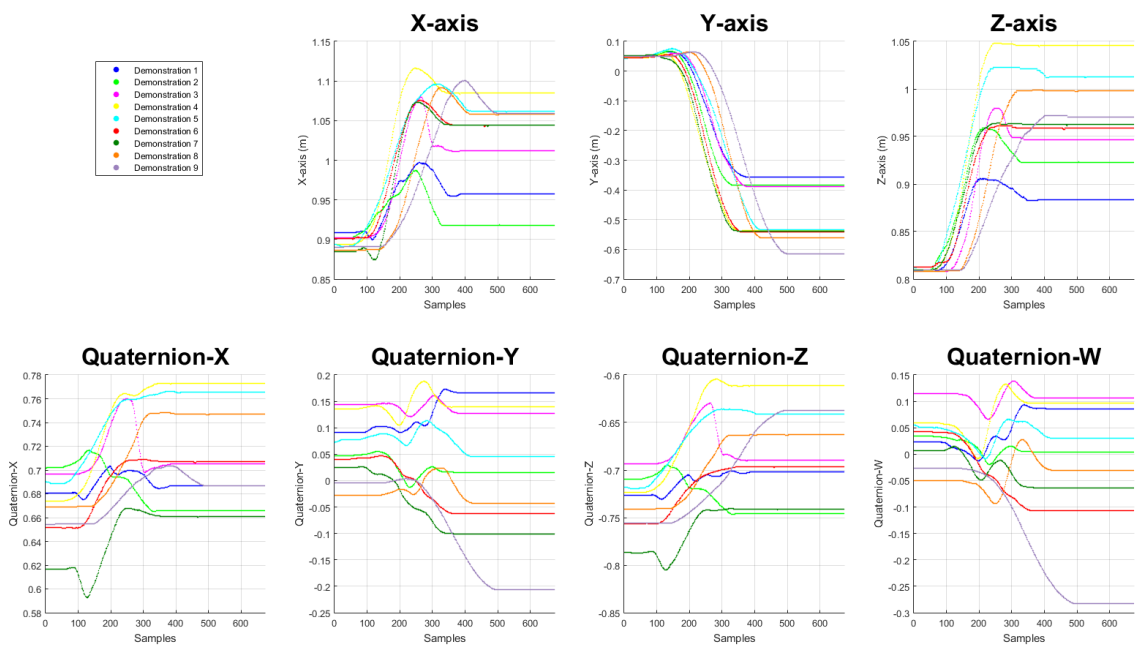


Figure 27: The sub-trajectories of the Right 3 group after Ramer-Douglas-Peucker simplification – Robot gripper assembly task.

The selected sub-trajectories for each group are processed by the Gaussian Mixture Model (GMM) method in 7-dimensions, as explained in section 3.3.2.2. In this chapter, the learned GMM for the manipulation of black part is presented, while the learned GMM for the manipulation of the other gripper's parts is shown in Appendix C.I. The learned GMM of the group Right 1, which represents the moving action of the right robotic arm from home to the grasping pose of the black part, is shown in Figure 28a for the *x-, y-* and *z-* dimensions (X-, Y-, Z-axis) and in Figure 29a for the quaternions. In like manner, the learned GMM of the groups Right 2 and 3 are shown in Figure 30a and Figure 32a for the *x-, y-* and *z-* dimensions (X-, Y-, Z-axis) and in Figure 31a and Figure 33a for the quaternions, respectively. The optimal number of Gaussians is equal to 6 for the groups Right 1-3.

- **Robot Working Phase (Online)**

During the robot working phase, two trials were performed. In the first trial, the pose of the gripper's parts differ from the demonstration. In the second trial, an obstacle was placed on the table, in a position within the demonstrated trajectory. The estimated pose of the parts by the *environmental perception module v1* during demonstration and during

two trials in working phase is shown in Table 12. In the presented task, the robot manipulated the gripper's parts, while the user was supposed to screw the parts together in order to successfully complete the assembly task. Figure 34a shows a real-time working environment (first trial), in which the user set up the gripper parts in poses different from the demonstrated ones.

Table 12: Estimated pose of objects by the *environmental perception module v1* during the demonstrations and the trials (working phase).

Objects	**X (m)**	**Y (m)**	**Z (m)**	**Rotation around Z-axis (°)**	
	Initial Scene during Demonstrations				
Top part	1.043	0.194	0.729	0.201	
Left side part	0.809	0.279	0.736	89.930	
Black part	1.041	-0.076	0.755	0.202	
Right side part	0.818	-0.196	0.736	90.200	
Base part	0.819	0.046	0.755	0.207	
Table	0.812	0.026	0.720	0.004	
Objects	**Initial Scene during Working Phase**				**Trial**
Top part	1.051	0.268	0.729	4.012	Trial 1 & 2
Left side part	0.785	0.279	0.736	-82.980	Trial 1 & 2
Black part	1.077	-0.084	0.755	6.004	Trial 1 & 2
Right side part	0.864	-0.221	0.736	89.920	Trial 1 & 2
Base part	0.811	0.052	0.755	0.205	Trial 1 & 2
Table	0.804	0.032	0.720	0.002	Trial 1 & 2
Red object (obstacle)	1.062	0.061	0.813	0.001	Trial 2

Firstly, the robot identified the task based on the objects in the scene. Subsequently, the robot confirmed that it was in the *Home* pose (which it was) and it suggested the next action, which offered the maximum reward (as explained in Algorithm 1 – section 3.4.1). However, the user was not satisfied with the suggested sequence of actions, which was the same as demonstrated. The reason was that the black part was heavy (approx. 1kg) and the vacuum did not provide a steady grasp, causing the part to oscillate. Therefore, the black part was not positioned properly, making the complete assembly to fail. The user decided to change the sequence of actions and the Q-Table converged in two iterations. The sequence of actions, learned by the reinforcement learning, had the following order:

1. manipulation of the left side part with the left robotic arm,
2. manipulation of the right side part with the right robotic arm,

3. manipulation of the top part with the left robotic arm, and

4. manipulation of the black part with the right robotic arm.

One remark is that Algorithm 1 (section 3.4.1) was updated to handle the robotic platform, consisting of two arms and multiple grippers. In case the gripper differed between two subsequently selected actions, then the m-GMM/GMR was skipped and the algorithm moved to the next state. For example, in Figure 24 it is shown that between the 4th (*LV End – Table info*) and 5th action (*RV Home – Home*), no moving action occurs.

After each action was confirmed by the user, the m-GMM/GMR algorithm was used to adapt the sub-trajectory to the new environmental conditions. In this chapter, the modification of the learned GMM for the manipulation of black part is presented, while the modification of the GMM for the manipulation of the other gripper's parts is shown in Appendix C.I. Table 12 shows that, during the working phase, the black part was in a different position along the X- and Y-axis as well as its orientation, in comparison to the demonstrated pose. Moreover, the base part was mounted on the table (as shown in Figure 34a) and the table was moved compared to the learning phase. The output of the m-GMM for the group Right 1, which represents the moving action of the right robotic arm from home to the grasping pose of the black part, is shown in Figure 28b for the *x-, y-* and *z-* dimensions (X-, Y-, Z-axis) and in Figure 29b for the quaternions. As it can be observed, the mean values of the Gaussians have been modified to accommodate the new pose of the black part. In like manner, the output of the m-GMM for the group Right 2, which is the moving action between the grasping of the black part and the placing of the black part next to the base, is shown in Figure 30b for the *x-, y-* and *z-* dimensions and in Figure 31b for the quaternions. As a result, the mean values of the Gaussians have been modified to satisfy the new poses of the black and base part. Furthermore, the m-GMM output of the group Right 3 is shown in Figure 32b for the *x-, y-* and *z-* dimensions and Figure 33b for the quaternions. Subsequently, the GMR method was used to generate the adapted sub-trajectory, which had to be followed by the robot. The generated GMR trajectory for the groups Right 1, 2, and 3, are presented in Figure 28c, Figure 30c, and Figure 32c for the *x-, y-* and *z-* dimensions and in Figure 29c, Figure 31c, and Figure 33c for the quaternions, respectively.

The robot performed the suggested sequence together with the human collaborator, as shown in Figure 34b. After each gripper's part manipulation was completed by the robot the human screws the parts together and then confirmed then next action of the robot. Videos of the robot's execution can be found in supplementary materials[8] in [1].

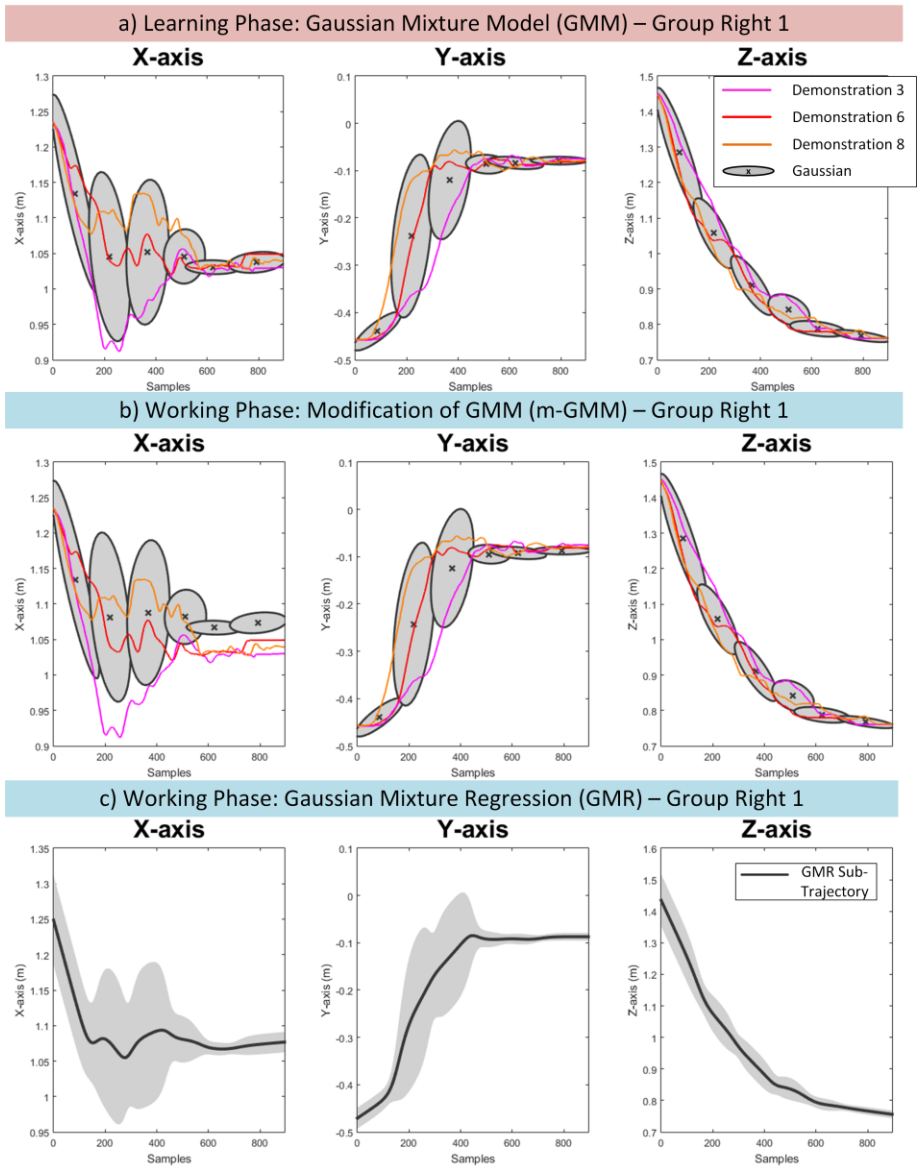


Figure 28: Robot gripper assembly task. A) The learned GMM for the sub-trajectories of group Right 1 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Right 1 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 1 along the X-, Y-, and Z-axis.

[8] The link directly to the supplementary materials in [1] is:

https://link.springer.com/article/10.1007/s10514-018-9725-6#SupplementaryMaterial

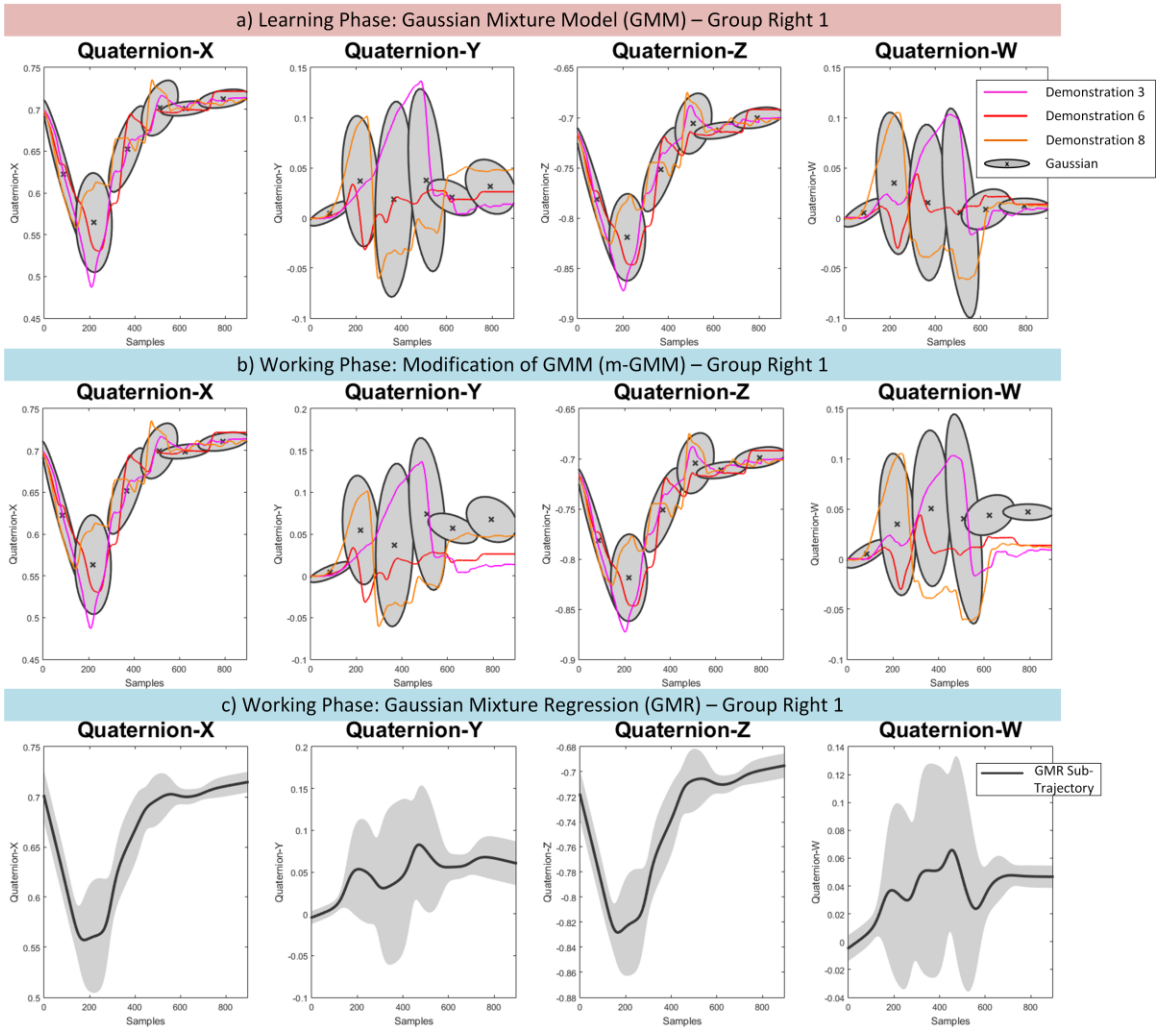


Figure 29: Robot gripper assembly task. A) The learned GMM for the sub-trajectories of group Right 1 for the quaternions. B) The modification of the learned GMM for the sub-trajectories of group Right 1 during the trial 1 for the quaternions. C) The generated GMR sub-trajectory produced by the m-GMM of group Right 1 for the quaternions.

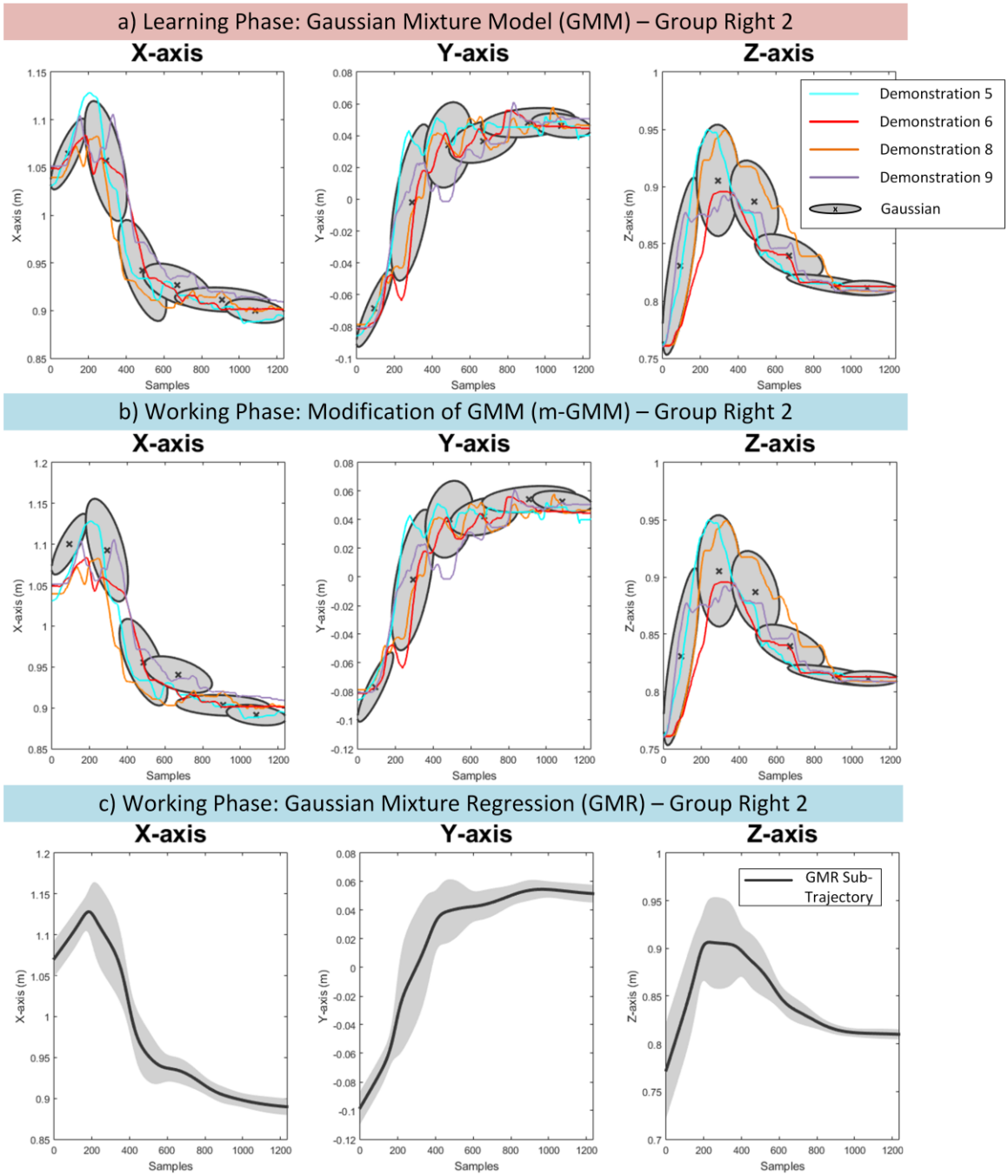


Figure 30: Robot gripper assembly task. A) The learned GMM for the sub-trajectories of group Right 2 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Right 2 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 2 along the X-, Y-, and Z-axis.

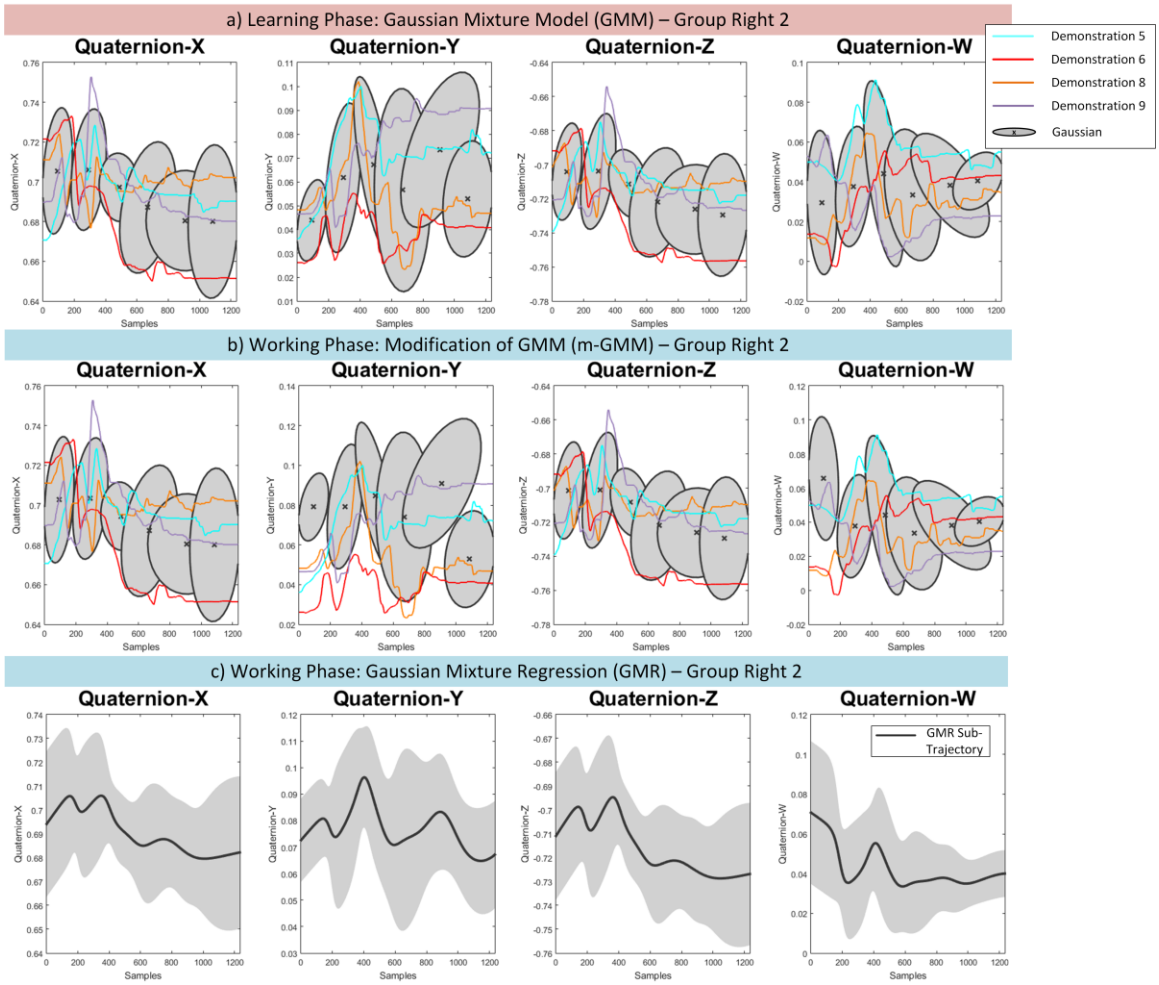


Figure 31: Robot gripper assembly task. A) The learned GMM for the sub-trajectories of group Right 2 for the quaternions. B) The modification of the learned GMM for the sub-trajectories of group Right 2 during the trial 1 for the quaternions. C) The generated GMR sub-trajectory produced by the m-GMM of group Right 2 for the quaternions.

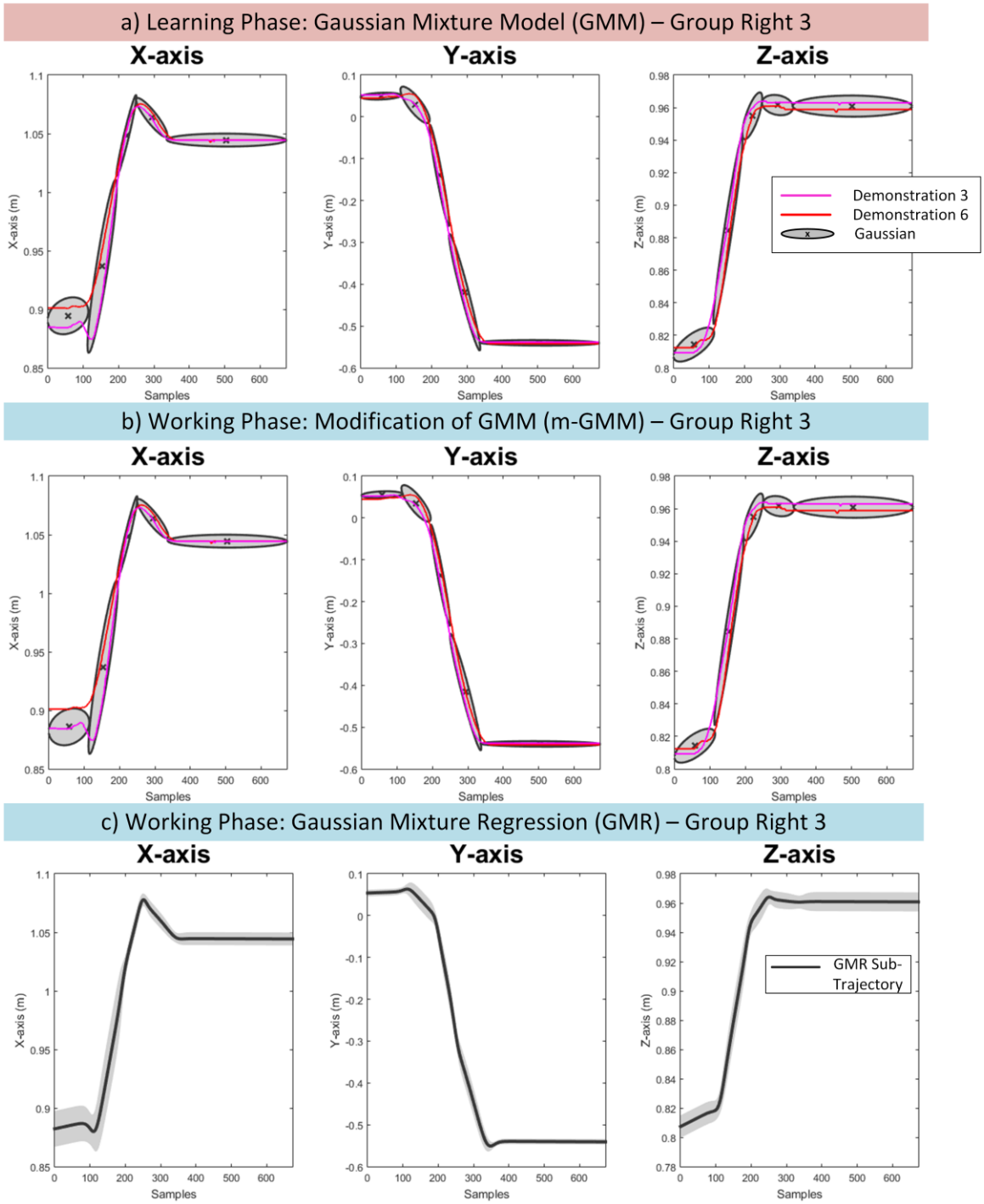


Figure 32: Robot gripper assembly task. A) The learned GMM for the sub-trajectories of group Right 3 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Right 3 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 3 along the X-, Y-, and Z-axis.

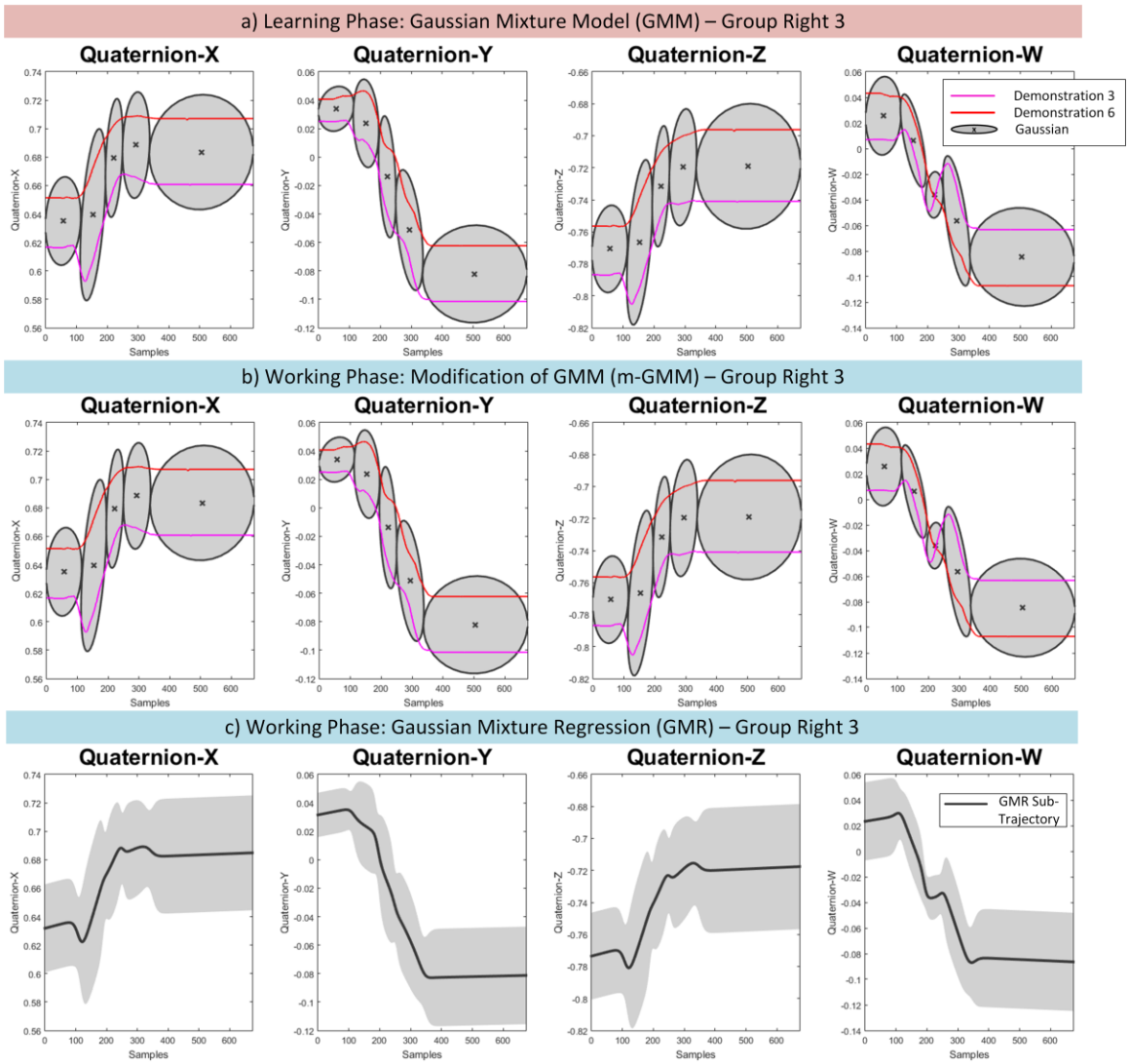


Figure 33: Robot gripper assembly task. A) The learned GMM for the sub-trajectories of group Right 3 for the quaternions. B) The modification of the learned GMM for the sub-trajectories of group Right 3 during the trial 1 for the quaternions. C) The generated GMR sub-trajectory produced by the m-GMM of group Right 3 for the quaternions.

Figure 34: Example of human–robot collaboration for the robot gripper assembly task during the trial 1. A) Initial scene during demonstrations (left) and during a real-time working environment (right), b) the robot executes the learned task with a different sequence of actions suggested by the human collaborator and different pose of gripper parts. (adapted from [1])

Figure 35 shows a real-time execution of the manipulation of the black part, in which an additional object (red object) is added as an obstacle during the second trial. The obstacle avoidance in m-GMM is illustrated in Figure 35b and it was calculated by adding, the height of the obstacle plus the height of the top part plus a safety distance to the mean values of the Gaussians in obstacle (as explained in Section 3.4.2.1).

The robot was able to avoid the obstacle but the picked object (black part) was not positioned properly, as it can be seen in Figure 35d. The reason is that the black part is heavy (1 kg) and it oscillated during the manipulation by the vacuum gripper which caused the object misplacement. Videos of the robot's execution can be found in supplementary materials[9] in [1].

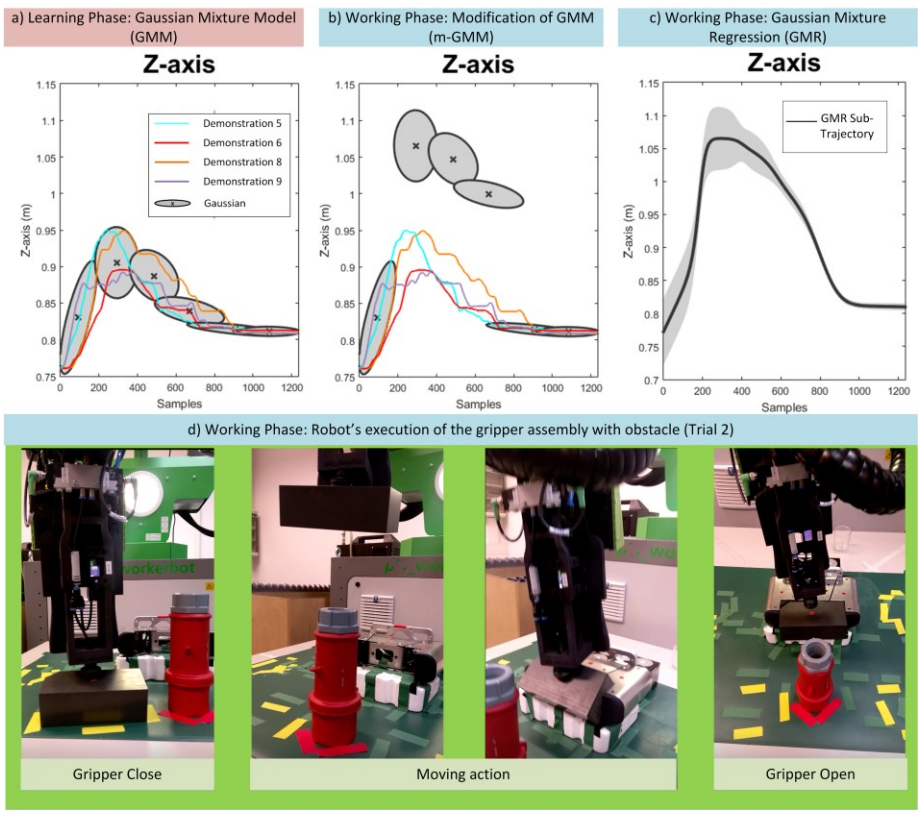


Figure 35: Example of the manipulation of the black part with obstacle avoidance along the Z-axis during trial 2. A) Learned GMM from the selected sub-trajectories, b) modified GMM to avoid obstacle, c) generated sub-trajectory via GMR from the modified GMM, d) the robot executes the manipulation of the black part. (adapted from [1])

[9] The link directly to the supplementary materials in [1] is:
https://link.springer.com/article/10.1007/s10514-018-9725-6#SupplementaryMaterial

4.3 'Pins into Holes' Task (Task 2)

To evaluate the proposed framework, a second industrial task was selected that requires high precision. The selected task 'Pins into Holes' (Task 2), illustrated in Figure 36, involves different objects, a color box with holes of different sizes and pins to be inserted in these holes. In Figure 36a-c, three versions of the Task 2 are shown where different pins are inserted in different holes of the color box. The first two versions of the Task 2 (Figure 36a, b) were demonstrated by users. To learn the necessary sequence of actions to perform the task, the presented RLfD was utilized. The Decision Support sub-module (section 3.4.1) was used to enable the robot to perform the third version of the Task 2 (Figure 36c) without additional training by the user. The robot used the knowledge gained from first two demonstrated versions of the task to perform an unseen version. The Figure 36d presents the pins used in the Task 2 and Figure 36e shows the initial scene of the task, where the pins are positioned in a wooden holder and a red holder is used to hold the color box.

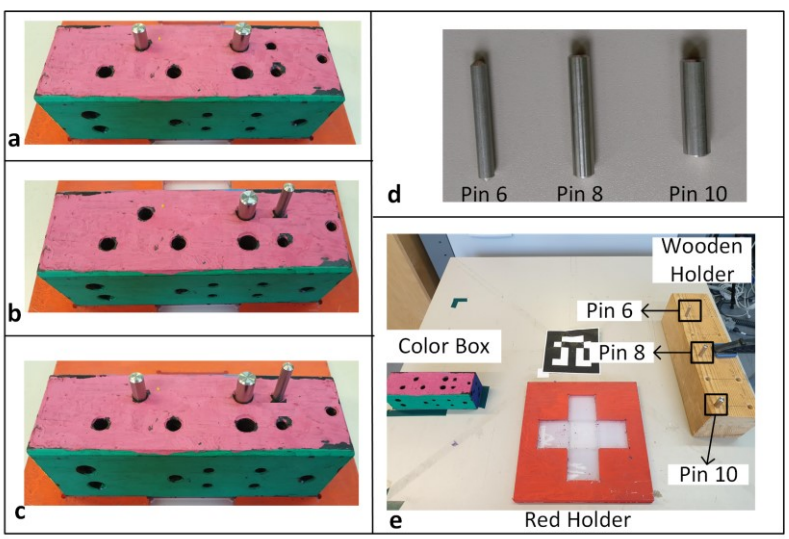


Figure 36: a), b) and c) Three versions of the 'Pins into Holes' task, d) Pins involved in the 'Pins into Holes' task, e) Initial scene of the 'Pins into Holes' task.

4.3.1 Human Demonstrations of 'Pins into Holes' Task

Nine participants were asked to provide demonstrations of the Task 2. The participants interacted with the Workerbot using a gamepad (section 3.2.1.2), as shown in Figure 37.

Out of 9 participants, 3 are females and 6 are males. The average age was 31.55±6.98 years old. All participants gave their informed, signed consent for the participation. All participants had ten minutes to get familiar with the gamepad, before they were asked to demonstrate the task.

Figure 37: A user demonstrates the task ‘Pins into Holes’ using a gamepad.

The participants demonstrated the first two versions of the Task 2 (Figure 36a, b). For the demonstrations only the pins involved in particular version were present in the initial scene. For example, for the initial scene of the first version (Figure 36a), pins 10 and 8 were placed in the wooden holder. The participants were informed about the goal scene for the two versions of the Task 2. The participants were allowed to choose the preferred sequence of actions (Figure 38) to achieve the goal. In Figure 38 the sequence of actions to perform the two versions of Task 2 is presented. For the first version of the Task 2 *pin X* could either be pin 8 or pin 10 and *pin Y* could either be pin 10 or pin 8 (*pin Y≠pin X*). For the second version *pin X* could either be pin 6 or pin 10 and *pin Y* could either be pin 10 or pin 6 (*pin Y≠pin X*). Each participant provided one demonstration for each version of the Task 2. In total, 18 demonstrations were recorded.

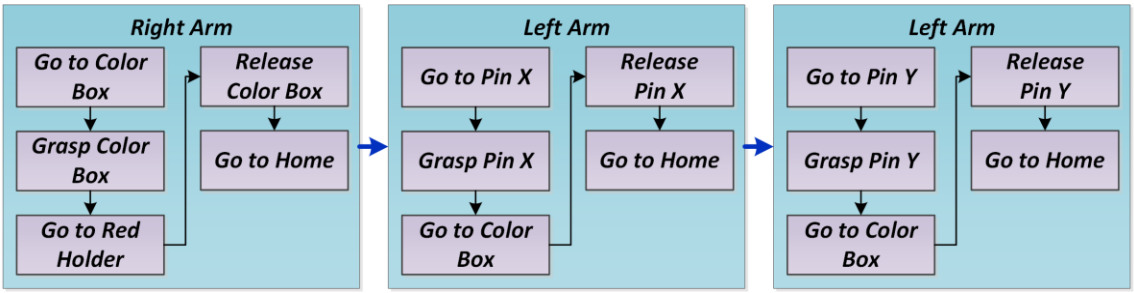


Figure 38: Sequence of actions to perform the first two versions of the 'Pins into Holes' task

Furthermore, a smaller study with two participants (two males of ages 24 and 27) was conducted, where 'hands-free' human-robot interaction using a motion sensor (section 3.2.2.1) was selected for the demonstration, as shown in Figure 39. All participants had twenty minutes prior to providing the demonstration to get familiar with the head gesture-based HRI.

In this study, a different concept was examined. The goal was that the robot had to learn the third version of the Task 2 (Figure 36c), which included all three pins. However, the robot already knew how to perform the second version of the task (Figure 36b), which included the manipulation of the color box and the pins 6 and 10. The user demonstrated to the robot how to manipulate the pin 8 and the robot learned the additional manipulation using a single demonstration (one-shot learning).

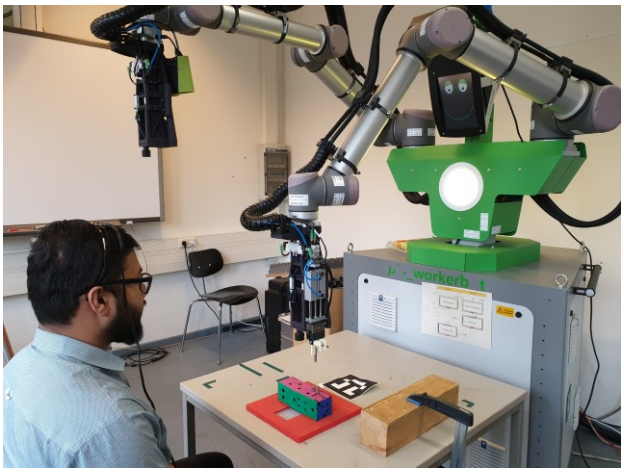


Figure 39: A user demonstrates a part of the 'Pins into Holes' task using head gestures.

4.3.2 Robot Learning and Working Phase of the 'Pins into Holes' Task

The next step after the demonstrations of Task 2 was the robot learning phase. Eighteen demonstrations were used as input to the robot learning phase.

- **Robot Learning Phase (Offline)**

The demonstrations were first processed by the Symbolic Task Learning (High-level) module, presented in section 3.3.1. The ATSSA module was used to generate the sequence of actions. However, not all the users followed the same sequence of actions. The sequence of actions generated by the ATSSA sub-module for the demonstrations is shown in Table 13.

Table 13: Output of ATSSA sub-module for the demonstrations of the task 'Pins into Holes'

	First Version of Task 2		**Second Version of Task 2**	
	5 Demonstrations	4 Demonstrations	4 Demonstrations	5 Demonstrations
1.	*RV Home – Home*	*RV Home – Home*	*RV Home – Home*	*RV Home – Home*
2.	*RV Close – Color box info*	*RV Close – Color box info*	*RV Close – Color box info*	*RV Close – Color box info*
3.	*RV Open – Red holder info*	*RV Open – Red holder info*	*RV Open – Red holder info*	*RV Open – Red holder info*
4.	*RV Home – Home*	*RV Home – Home*	*RV Home – Home*	*RV Home – Home*
5.	*L2 Home – Home*	*L2 Home – Home*	*L2 Home – Home*	*L2 Home – Home*
6.	*L2 Close – Pin 10 info*	*L2 Close – Pin 8 info*	*L2 Close – Pin 10 info*	*L2 Close – Pin 6 info*
7.	*L2 Open – Color box info*	*L2 Open – Color box info*	*L2 Open – Color box info*	*L2 Open – Color box info*
8.	*L2 Home – Home*	*L2 Home – Home*	*L2 Home – Home*	*L2 Home – Home*
9.	*L2 Close – Pin 8 info*	*L2 Close – Pin 10 info*	*L2 Close – Pin 6 info*	*L2 Close – Pin 10 info*
10.	*L2 Open – Color box info*	*L2 Open – Color box info*	*L2 Open – Color box info*	*L2 Open – Color box info*
11.	*L2 Home – Home*	*L2 Home – Home*	*L2 Home – Home*	*L2 Home – Home*

The position of the objects and the table during the demonstrations were estimated with respect to the world coordinate system by the *environmental perception module v1* and the results are shown in Table 14. The initialization of the Q-table (section 3.3.1.2) for the demonstrated Task 2 is shown in Table 15.

Table 14: Estimated pose of objects by the *environmental perception module v1* during the demonstrations of the task 'Pins into Holes'.

Object	**Initial Scene – Position**			**Present in the initial scene of the Task 2 version**
	X (m)	**Y (m)**	**Z (m)**	
Color box	0.946	-0.257	0.752	Version 1 & 2
Pin 10	0.960	0.245	0.822	Version 1 & 2
Pin 8	0.850	0.245	0.827	Version 1
Pin 6	0.740	0.245	0.827	Version 2
Red holder	1.040	0.043	0.730	Version 1 & 2
Wooden holder	0.737	0.277	0.769	Version 1 & 2
Table	0.804	0.032	0.720	Version 1 & 2

Table 15: Initialization of the Q-Table for the demonstrated task 'Pins into Holes'.

	Actions a							
States g	*RV Home – Home*	*RV Close – Color box info*	*RV Open – Red holder info*	*L2 Home – Home*	*L2 Close – Pin 10 info*	*L2 Open – Color box info*	*L2 Close – Pin 8 info*	*L2 Close – Pin 6 info*
1	**1**	0	0	0	0	0	0	0
2	0	**1**	0	0	0	0	0	0
3	0	0	**1**	0	0	0	0	0
4	**1**	0	0	0	0	0	0	0
5	0	0	0	**1**	0	0	0	0
6	0	0	0	0	**0.5**	0	**0.22**	**0.28**
7	0	0	0	0	0	**1**	0	0
8	0	0	0	**1**	0	0	0	0
9	0	0	0	0	**0.5**	0	**0.28**	**0.22**
10	0	0	0	0	0	**1**	0	0
11	0	0	0	**1**	0	0	0	0

The demonstrations were further processed by the Skill Learning at Trajectory Level (Low-level) module, presented in section 3.3.2. The demonstrated trajectories of the left and right robotic arms split into $G_L - 1$ and $G_R - 1$ sub-trajectories, respectively. The next step was to group the sub-trajectories, resample them and select similar sub-trajectories, as explained in section 3.4.2.1. Table 16 shows the association between the high-level actions and the groups of sub-trajectories. The groups Right 1 – 3 and Left 5 – 7 consist of eighteen demonstrations, while the remaining groups consist of nine

demonstrations. This happens because the color box and the pin 10 were manipulated in both versions of the Task 2.

The selected demonstrations were further processed by the GMM method (section 3.4.2.2). The results of the learned GMM for the manipulation of the pin 6 are presented in this chapter. The results for the manipulation of the other pins and the color box are shown in Appendix C.II. Figure 40a shows the learned GMM for the group Left 8 – version 2 for the x- , y- and z-dimensions (X-, Y-, Z-axis). The group Left 8 – version 2 is the moving action between the high-level actions *L2 Home – Home* and *L2 Close – Pin 6 info*, as shown in Table 16. Similarly, Figure 41a and Figure 42a show the learned GMM for the group Left 9 – version 2 and Left 10 – version 2, respectively. The optimal number of Gaussians is equal to 5, 4, 3 for the groups Left 8 –, 9 –, and 10 – version 2, respectively. The orientation of the objects did not change between learning and working phase. Therefore, only the *x-, y-* and *z-* dimensions are shown in this chapter.

Table 16: Association between the groups of sub-trajectories and high-level actions for the first and second version of the task 'Pins into Holes' (Task 2).

First Version of Task 2		**Second Version of Task 2**	
Group	**Sub-trajectories between the high-level actions**	**Group**	**Sub-trajectories between the high-level actions**
Right 1	*RV Home – Home* and *RV Close – Color box info*	Right 1	*RV Home – Home* and *RV Close – Color box info*
Right 2	*RV Close – Color box info* and *RV Open – Red holder info*	Right 2	*RV Close – Color box info* and *RV Open – Red holder info*
Right 3	*RV Open – Red holder info* and *RV Home – Home*	Right 3	*RV Open – Red holder info* and *RV Home – Home*
Left 5	*L2 Home – Home* and *L2 Close – Pin 10 info*	Left 5	*L2 Home – Home* and *L2 Close – Pin 10 info*
Left 6	*L2 Close – Pin 10 info* and *L2 Open – Color box info*	Left 6	*L2 Close – Pin 10 info* and *L2 Open – Color box info*
Left 7	*L2 Open – Color box info* and *L2 Home – Home*	Left 7	*L2 Open – Color box info* and *L2 Home – Home*
Left 8 – version 1	*L2 Home – Home* and *L2 Close – Pin 8 info*	Left 8 – version 2	*L2 Home – Home* and *L2 Close – Pin 6 info*
Left 9 – version 1	*L2 Close – Pin 8 info* and *L2 Open – Color box info*	Left 9 – version 2	*L2 Close – Pin 6 info* and *L2 Open – Color box info*
Left 10 – version 1	*L2 Open – Color box info* and *L2 Home – Home*	Left 10 – version 2	*L2 Open – Color box info* and *L2 Home – Home*

- **Robot Working Phase (Online)**

During the robot working phase, the robot was required to perform the third version of the Task 2 (Figure 36c) without additional training by the user. Moreover, to evaluate the learning of user's preferences, two users interacted with the robot. The initial scene consisted of all 3 pins (Figure 36e) and the positions of the pins are shown in Table 17. Firstly, the robot identified the task based on the objects in the scene as Task 2 and the Q-Table (initialized by the demonstrations) was loaded. Subsequently, the robot confirmed that it was in the *Home* pose and it suggested the next action, which offered the maximum reward (as explained in Algorithm 1 – section 3.4.1).

The robot suggested the following sequence of high-level actions: 1. Manipulation of the color box, 2. Manipulation of the pin10, and 3. Manipulation of the pin 10. The reason that the manipulation of the pin 10 was suggested twice is because the reward is the highest (Table 15 – state 9). The user 1 confirmed the high-level actions for the first 2 manipulations. However, the user rejected the proposed high-level actions for third manipulation, and the robot suggested the manipulation of the pin 8, which was rejected again by the user. At the end, the user confirmed the last suggestion by the robot, which was the manipulation of the pin 6. After the manipulation of the pin 6, the user added additional states and selected the manipulation of pin 8 (Algorithm 1 – section 3.4.1). In the same fashion, the user 2 confirmed the following sequence of high-level actions: 1. Manipulation of the color box, 2. Manipulation of the pin 6, 3. Manipulation of the pin 8, 4. Manipulation of the pin 10. For each user, the Q-Table converged after two to four iterations and the suggestions from the robot were accepted by the users.

Table 17: Estimated pose of objects by the *environmental perception module v1* during the demonstrations of the task 'Pins into Holes'.

Object	**Initial scene during learning phase**			**Initial scene during working phase**		
	X (m)	**Y (m)**	**Z (m)**	**X (m)**	**Y (m)**	**Z (m)**
Color box	0.946	-0.257	0.752	0.951	-0.258	0.752
Pin 10	0.960	0.245	0.823	1.092	0.343	0.823
Pin 8	0.850	0.245	0.827	0.982	0.287	0.827
Pin 6	0.740	0.245	0.827	0.872	0.343	0.827
Red holder	1.040	0.043	0.730	1.041	0.045	0.730
Wooden holder	0.737	0.277	0.769	0.869	0.319	0.769
Table	0.804	0.032	0.720	0.805	0.034	0.720

After each action was confirmed by the user, the m-GMM/GMR algorithm was used to adapt the sub-trajectory to the new environmental conditions. In this chapter, the modification of the learned GMM for the manipulation of pin 6 is presented, while the m-GMM/GMR of other pins and the color box is presented in Appendix C.II. The pin 6 was the most challenging to be manipulated as it was the thinnest pin. Table 17 shows that during the working phase the pin 6 was in a different position along the X- and Y-axis, in comparison to the demonstrated pose. Furthermore, the red holder, on which the color box should be placed, was mounted on the table (as shown in Figure 36e) and the table was moved compared to the learning phase. The output of the m-GMM for the group Left 8 – version 2, which represents the moving action of the left robotic arm from home to the grasping pose of the pin 6, is shown in Figure 40b for the *x-, y-* and *z-* dimensions. As it can be observed, the mean values of the Gaussians have been modified based on the new position of the pin 6. Moreover, the output of the m-GMM for the group Left 9 – version 2, which is the moving action between the grasping of the pin 6 and the placing of it into the color box, is presented in Figure 41b. As a result, the mean values of the Gaussians have been modified to satisfy the new positions of the pin 6 and the color box, which is placed on the red holder. Furthermore, the m-GMM output of the group Left 10 – version 2, is shown in Figure 42b. Subsequently, the GMR method was used to generate the adapted sub-trajectory, which had to be followed by the robot. The generated GMR trajectory for the groups Left 8 –, 9 –, and 10 – version 2, are presented in Figure 40c, Figure 41c, Figure 42c respectively. Figure 43 illustrates the robot execution of the 'Pins into Holes' version 3 Task, with the sequence of actions preferred by the user 1.

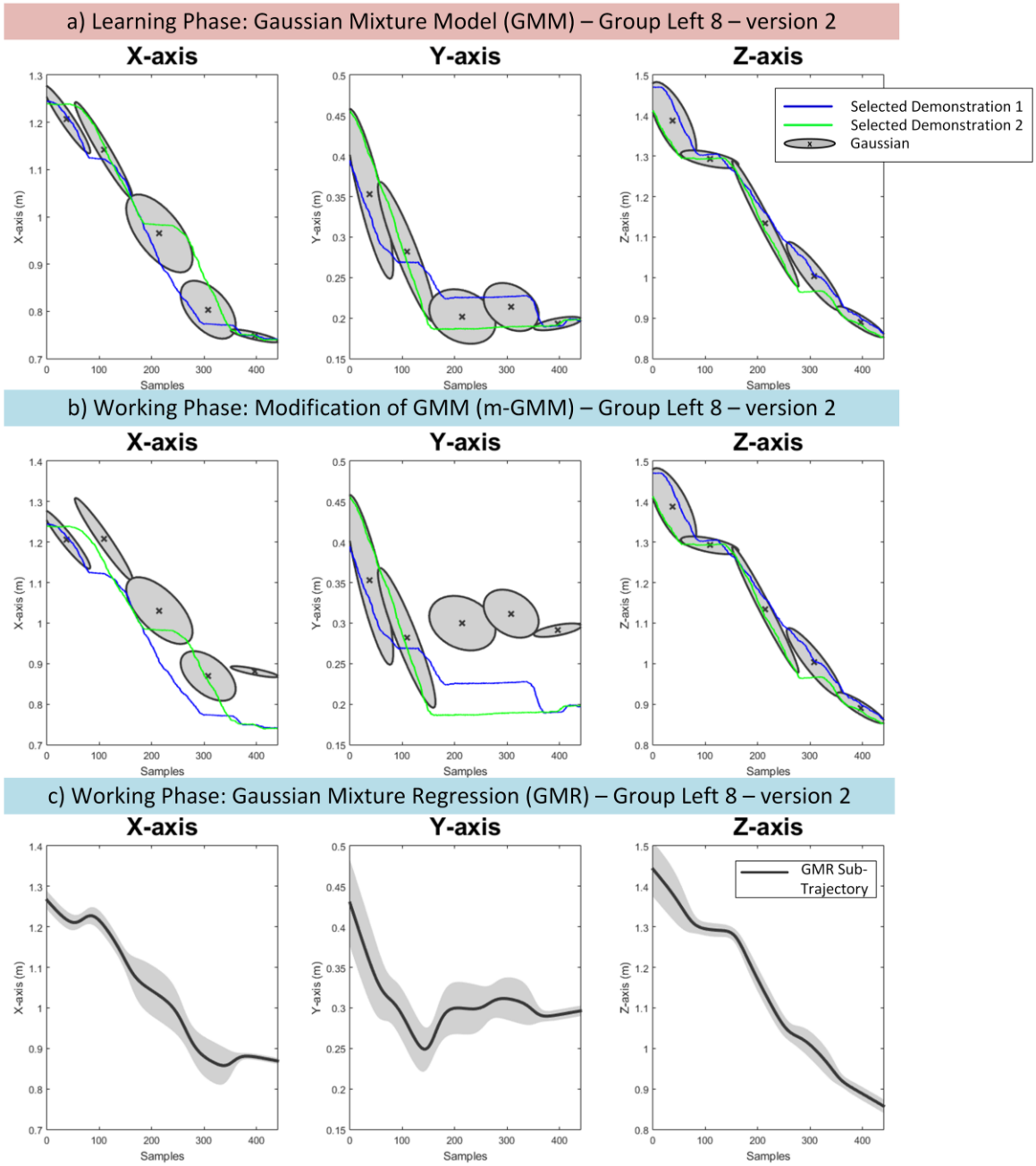


Figure 40: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Left 8 – version 2 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 8 – version 2 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 8 – version 2 along the X-, Y-, and Z-axis.

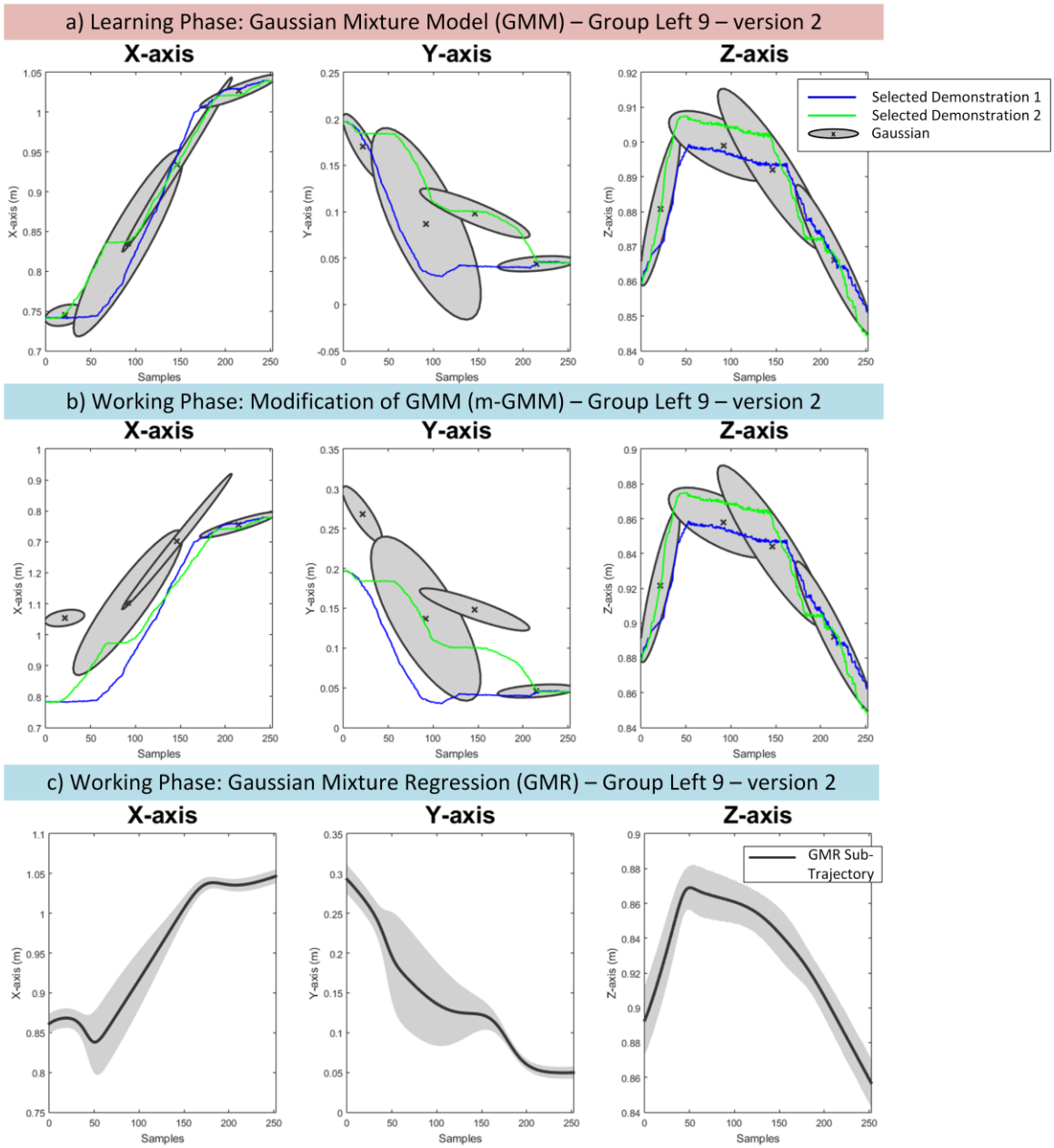


Figure 41: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Left 9– version 2 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 9 – version 2 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 9 – version 2 along the X-, Y-, and Z-axis.

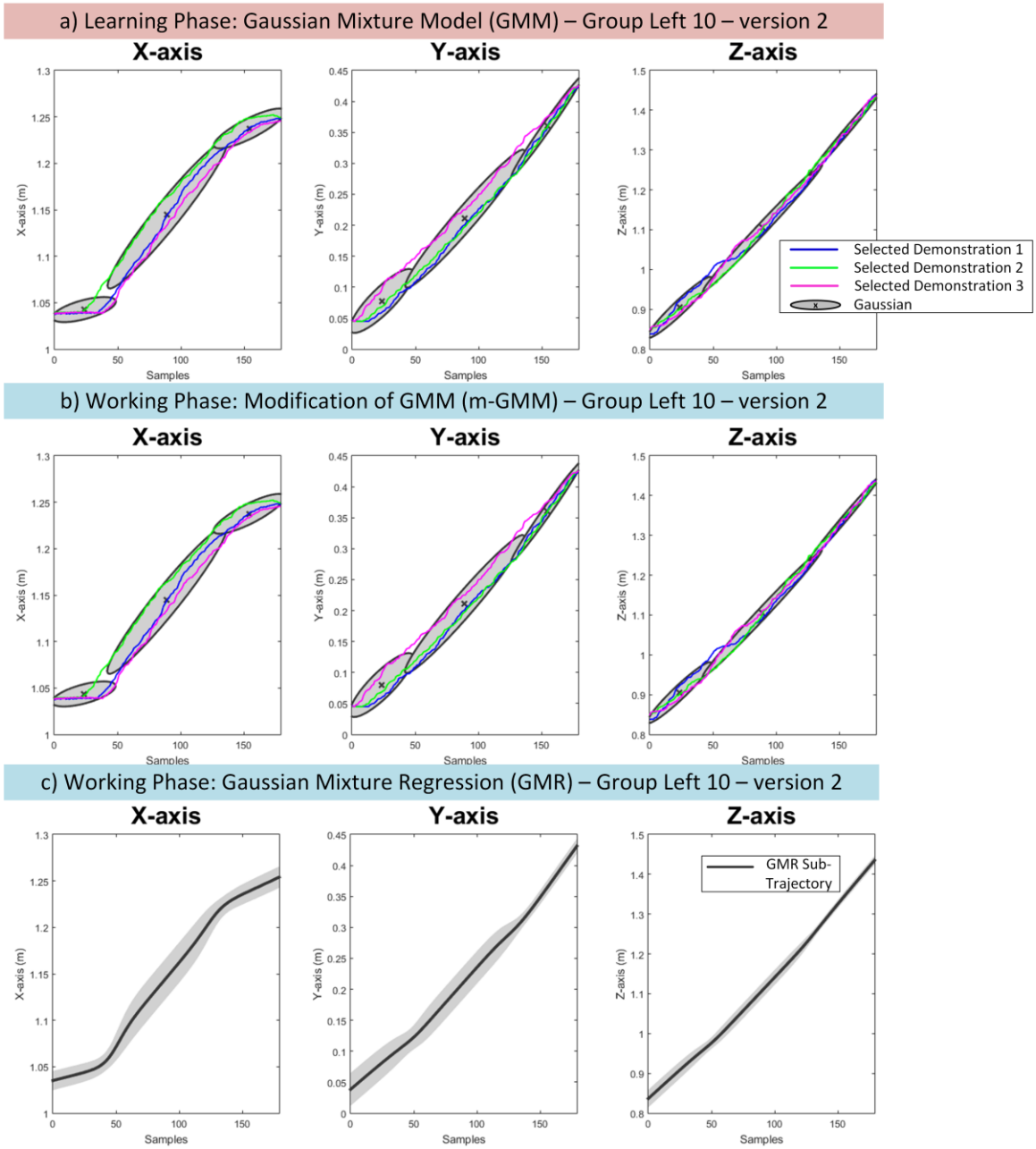


Figure 42: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Left 10– version 2 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 10 – version 2 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 10 – version 2 along the X-, Y-, and Z-axis.

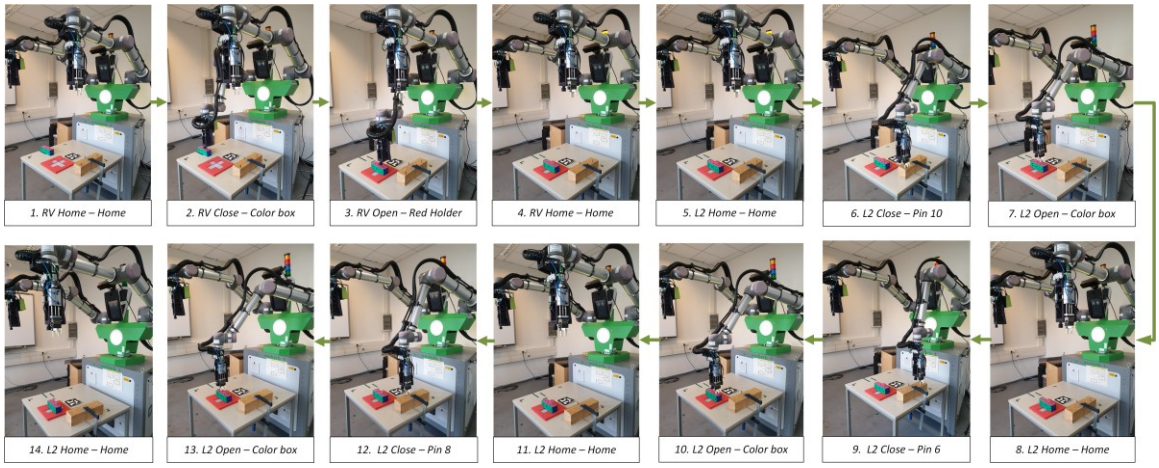


Figure 43: Robot execution of the task 'Pins into Holes' version 3 with the preferable sequence of actions for the user 1

Moreover, a different concept was examined during the working phase. The robot had only learned the second version of the Task 2 (Figure 36b), i.e. how to manipulate the color box and the pin 6 and 10. However, there were three pins in the initial scene as shown in Figure 36e. The user had to teach the robot to manipulate the third pin (pin 8) using the head gestures-based HRI. The robot had to learn the third version of the Task 2 (Figure 36c), which included all three pins. Nevertheless, the complete manipulation of pin 8 using head gestures is very challenging and time consuming as it needs high precision. Since the robot already knew how to manipulate pins of other sizes, the moving actions from home to grasping point of a pin and from releasing a pin to home could be easily modified using the m-GMM to manipulate the pin 8. In other words, the learned GMM between home and close action and between open and home action of the pin 10 or pin 6 can be modified for the pin 8 using the m-GMM. However, the robot had no knowledge of where the pin 8 had to be placed. The user guided the robot though the moving action between grasping of the pin 8 and releasing it into the proper hole of the color box. The robot learned the additional moving action using a single demonstration (one-shot learning).

The first user needed 430 seconds (7.2 minutes) and 73 gestures to demonstrate the missing moving action and the second user needed 480 (8 minutes) and 76 gestures. The learned GMM for the demonstrated moving action of pin 8 from the first user is shown in Figure 44a for the x- , y- and z-dimensions. The GMR that is reproduced by the learned

GMM is shown in Figure 44b and it can be observed that the generated GMR is similar to the demonstration. The user did not change the orientation of the robot's end-effector during the demonstration.

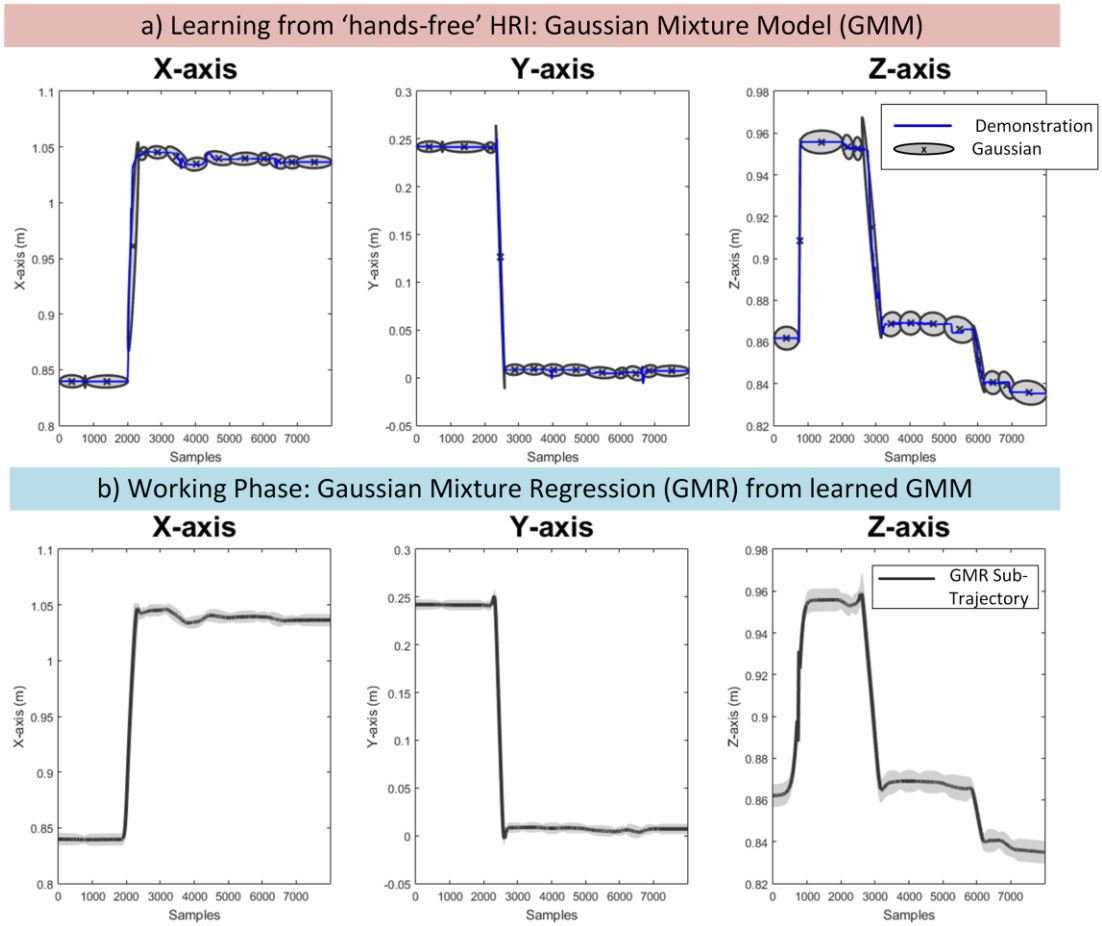


Figure 44: Learning from 'hands-free' HRI for task 'Pins into Holes'. a) The learned GMM for the demonstrated sub-trajectory along the X-, Y-, and Z-axis. b) The generated GMR sub-trajectory produced by the learned GMM along the X-, Y-, and Z-axis.

4.4 Discussion

In this chapter the presented RLfD framework utilizes a dual-arm industrial robot to learn and reproduce assembly tasks. The first human-robot synergetic task was the assembly of a robot gripper, and the second challenging task was to insert 'Pins-into-Holes'. The presented results confirm the potential of the developed RLfD framework to empower robots with learning abilities. The robot was able to reproduce the demonstrated tasks in changed environment, to suggest sequence of high-level actions, to exploit prior

knowledge to execute an unseen version of the task (version 3 of Task 2), and to learn user's preferences. Different HRI methods were employed for the demonstrations and the RLfD was able to successfully learn the demonstrated tasks.

Two participants (a male and a female), who provided demonstrations for both tasks using hand-operated HRIs, stated that the kinesthetic teaching approach was easier to learn in comparison to the gamepad. However, they pointed out that they felt physical tiredness on their arms due to the weight of the robotic arms. Although the robotic arms were in gravity-compensation mode during the kinesthetic teaching, they felt tired after some time.

Nevertheless, it is important to state the RLfD framework relies on the human demonstrations, and the learned trajectories depend on the quality of the demonstrations. In the presented RLfD framework, a first step is made by selecting similar demonstrations to be used as input for the robot learning at trajectory level. In future, a measurement to identify the quality of a demonstration is required and only demonstrations with high quality should further be processed by the RLfD framework.

Furthermore, it is observed by the presented results that the generated trajectories are not optimal. One approach to optimize the trajectories is to modify the covariance matrices of the GMM. However, if the covariance matrices change, it is not guaranteed that the constraints of the demonstrated trajectories are fulfilled. In future, a method that optimizes the GMM/GMR output is required, which will respect the constraints of the demonstrated trajectories.

Moreover, a 'hands-free' HRI concept is also presented in this chapter. Although the results show the potential of the 'hands-free' HRI to be used for teaching additional actions in an already learned task, the time and effort required by the user are high. One approach to overcome this problem would be to further segment moving actions. In other words, it will require the robot to recognize more detailed moving actions, such as moving close to an object, moving away from an object etc. For example, in the presented 'hands-free' HRI task, the robot would be able to additionally move the pin 8 away from the wooden holder, and then close to the color box. The user would only be required to guide the robot for a short distance to insert the pin into the right hole.

However, the objects would be required to be categorized to enable the robot to understand how to manipulate an unseen object with similar characteristics to a known manipulated object.

5. Robot Learning of an Assistive Manipulation Task from One-shot Human Demonstration – Application in Assistive Robotics

In this chapter, the presented robot learning from one-shot human demonstration is evaluated for an application in assistive robotics. Using a 'hands-free' HRI, the user is able to teach the robot a desired manipulation task. After the demonstration and the learning, the robot performs the task, even when the positions of the cup and the human have changed (in comparison to the demonstration). To demonstrate the feasibility of the RLfD framework, a small study was conducted with 13 participants; 12 able-bodied and 1 tetraplegic suffering from multiple sclerosis. The results have been presented by Kyrarini et al. in [4]. The study was performed in a real-world assistive scenario, which is explained in detail as follows.

5.1 Assistive Robotic Platform

The KINOVA JACO Gen2 [147] was selected as an assistive robotic platform, which is a 7-DoF ultra-lightweight robotic arm with a three-finger gripper attached as the end-effector. The JACO arm was mounted on a table, as shown in Figure 45. The JACO arm has been specifically designed as an assistive robot. Studies [148] [149], which have been

conducted with the JACO and individuals with upper-extremity disability, indicated the efficacy of JACO arm as an alternative to increase the autonomy of motor impaired users.

As a vision sensor the lightweight Intel Realsense Camera D435 [150] was selected to provide environmental information and was mounted on the robot's end effector between the gripper and the last joint using a 3D-printed attachment clamp, as illustrated in Figure 45. The Realsense contains a color camera (RGB) and a depth camera system, comprising two Infrared (IR) cameras and an IR projector [151].

The JACO arm was controlled by the head gesture-based HRI using a vision sensor mounted on a hat, which is presented in section 3.2.2.2. Figure 45 shows the setup of the assistive robotic platform and a user suffering from tetraplegia, who controlled the robot by head gestures. Figure 46a shows the world coordinate system, which is located at the robot's base.

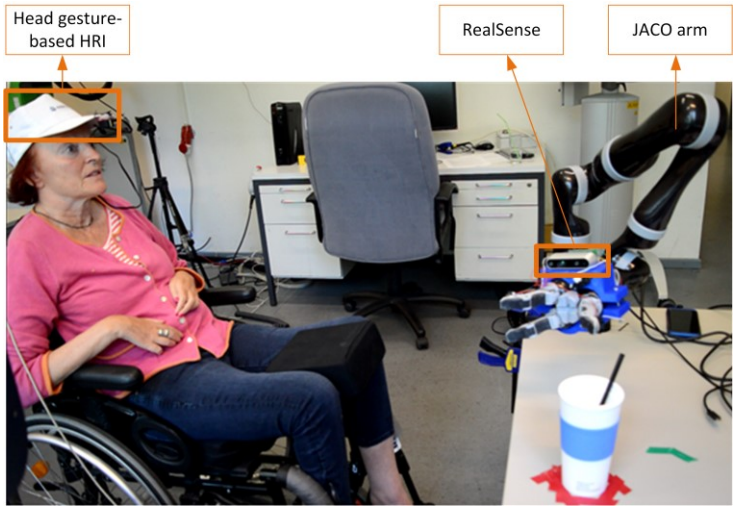


Figure 45: Setup of the presented system with the tetraplegic end-user

5.2 Assistive Manipulation Task for Serving a Drink

Assistive robotic manipulators have the potential to support individuals with tetraplegia in regaining their independence in performing Activities of Daily Living (ADLs). In the pre-development survey with potential end-users of robotic manipulators [152], it was shown that drinking, eating and preparing meals are highly prioritized tasks. To evaluate the presented RLfD framework, the assistive manipulation task to serve a drink was selected and it is described as follows; the user controlled the robot to pick a

cup from the table and to bring the cup close to their mouth to drink. Figure 46 illustrates the sequence of actions needed to complete the task.

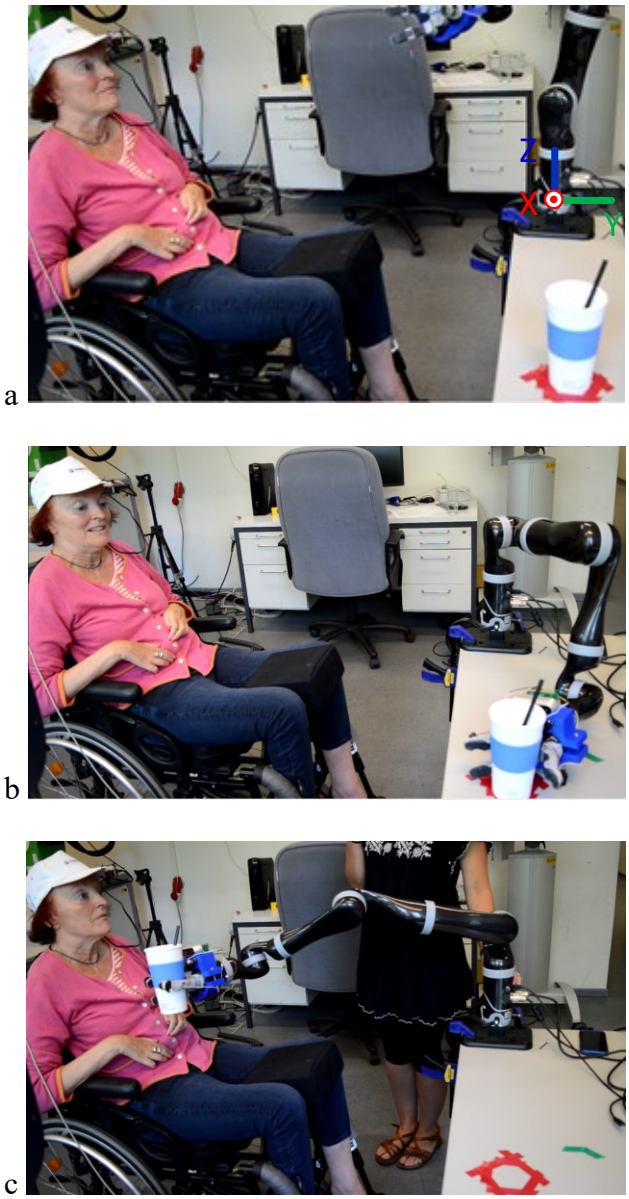


Figure 46: The sequence of actions for the assistive manipulation task. A) Robot is in the 'home' pose and the world coordinate system is marked, b) The user controls the robot to grasp the cup, c) The user brings the cup close to her.

5.3 Human Demonstration of the Assistive Manipulation Task

A small study was conducted, in which participants were asked to perform the assistive manipulation task, shown in Figure 46. Thirteen participants (12 able-bodied and 1 tetraplegic) took place in the study to perform one demonstration. Out of 13

participants, 4 are females (3 able-bodied and 1 tetraplegic) and 9 are males. The average age was 30.4±9.7 years old. All participants gave their informed, signed consent to participate in this study.

All the participants, who took place in the study, were able to learn how to provide demonstration of the task effectively and to demonstrate the task successfully in the first attempt. The average time to demonstrate the complete assistive manipulation task for 13 participants was 754.67±223.13 sec, while the tetraplegic user needed 738sec. The average number of head gestures to control successfully the robot through the task was 76.15±17.68, while the tetraplegic user needed 72 gestures. Obviously, the time taken by the tetraplegic user for providing the demo and the total number of gestures were comparable to the able-bodied participants. Furthermore, the participants reported their feedback on a questionnaire based on a 5-point Likert scale [153]. The questions were agreement-based and the Likert scale ranges from 1 ("strongly disagree") to 5 ("strongly agree"). The results are shown in Figure 47.

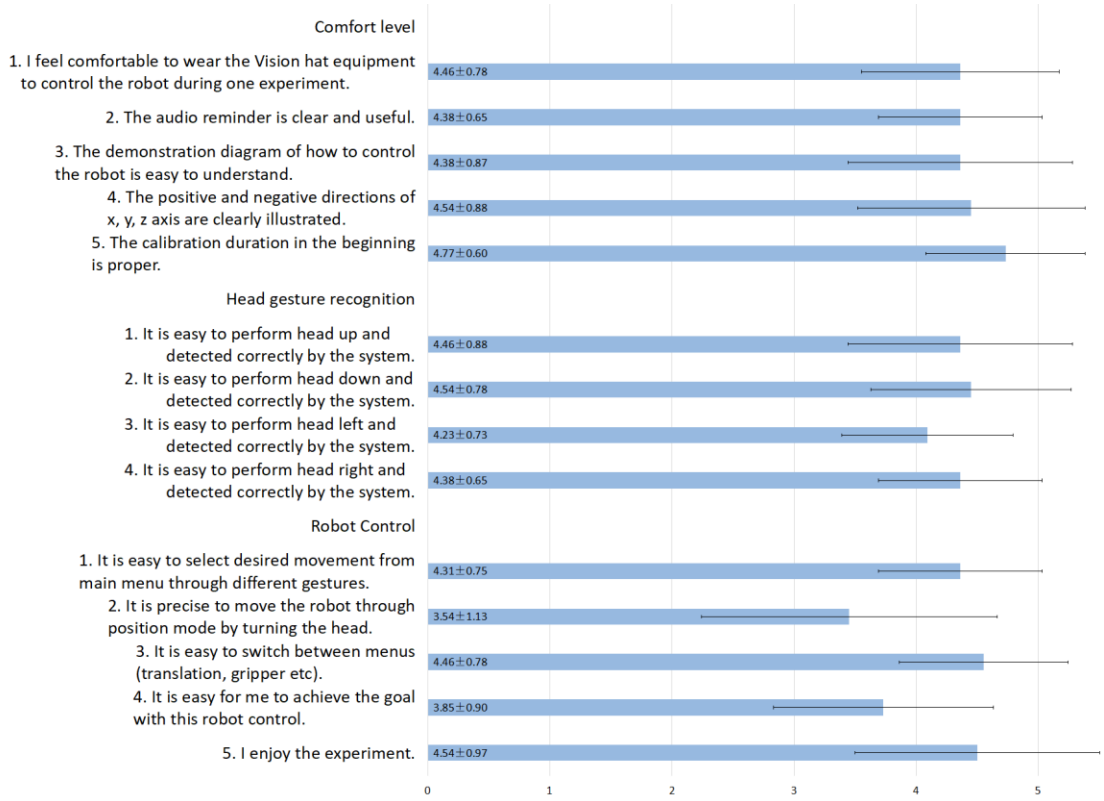


Figure 47: Subjective user feedback on the questionnaire based on a 5-point Likert scale [4].

Four of the participants (2 males and 2 females), including the participant with tetraplegia, had previous experience with head gesture-based HRI [154] [5]. The robot's end-effector trajectories demonstrated by the four experienced users are presented in Figure 48. It can be seen that each user selected a different path to guide the robot through the task. Moreover, the grasping preference of the cup for each user was different, even if the cup was placed at the same position though all the demonstrations. In detail, some users preferred to grasp the cup closer to its top edge, while others closer to its bottom edge. Some participants also preferred to grasp the cup having the complete fingers of the gripper in contact with the cup, while others only the fingertips. Furthermore, the position of the grasped cup as ready for drinking varies as humans vary in shape and height. Also, during the study, the tetraplegic user was seated on a wheelchair, while the able-bodied participants on a normal chair.

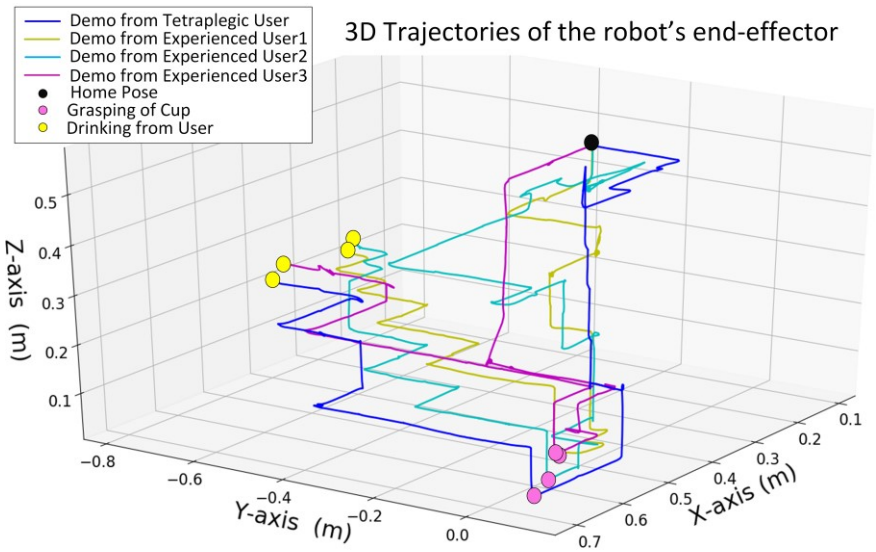


Figure 48: The trajectories of the end-effector in X, Y, Z-axis (3D) demonstrated by four participants having previous experience on head gesture-based human-robot interaction

5.4 Robot Learning and Working Phase of the Assistive Manipulation Task

The next step, after a user demonstrated the assistive manipulation task, was the robot learning (one-shot). In this thesis, the processing of the tetraplegic user is presented, as the presented RLfD has the potential to support individuals with tetraplegia in regaining

their independence in performing ADLs. However, the same procedure was performed for the demonstrations of the other users.

- **Robot Learning Phase (Offline)**

The demonstration was first processed by the Symbolic Task Learning (High-level) module, presented in section 3.3.1. The sequence of actions generated by the ATSSA sub-module was:

1. Home – Home,

2. Close – Cup info,

3. End – Person info.

The position of the cup and the person (user) during the demonstration was estimated with respect to the world coordinate system by the *environmental perception module v2* and the results are shown in Table 18. Based on equation (3.1) and (3.2), and the output of the ATSSA sub-module, the initialization of the Q-table for the demonstrated manipulation task is shown in Table 19.

Table 18: Estimated position of the person and the cup during the demonstration by the *environmental perception module v2*

Obj.	**X (m)**	**Y (m)**	**Z (m)**
	During Demonstration		
Person	0.425	-0.773	0.372
Cup	0.611	0.059	0.050

Table 19: Initialization of the Q-Table for the demonstrated manipulation task

	Actions *a*		
States *g*	*Home – Home*	*Close – Cup info*	*End – Person info*
1	**1**	0	0
2	0	**1**	0
3	0	0	**1**

Second, the demonstrated trajectory was split into two sub-trajectories. The first sub-trajectory was from *Home* pose to the grasping point of the cup (*Close*) and the second from grasping point of the cup to the drinking point for the user (*End*). Figure 49a-c and Figure 50a-c respectively show the first and second sub-trajectory of the demonstration provided by the tetraplegic user and the learned GMM for the *x-, y-* and *z-* dimensions (X-, Y-, Z-axis). The optimal number of Gaussians was equal to 12 for the first sub-

trajectory and 15 for the second. The orientation of the cup and the person did not change between learning and working phase. Therefore, only the *x-, y-* and *z-* dimensions are shown in this chapter. In Appendix D, all 7 dimensions of the learned GMM are shown for the two sub-trajectories.

- **Robot Working Phase (Online)**

To evaluate the m-GMM algorithm, developed in this thesis, three trials were performed in robot working phase. In each trial, the position of the person and the cup differ from the demonstration, and the robot started from the *Home* pose. In the third trial, a cubic box of dimensions 20x20x20 cm was positioned on the table, in a position within the demonstrated trajectory. The estimated position of the cup and the person by the *environmental perception module v2* during demonstration and during three trials in working phase is shown in Table 20.

Table 20: Estimated position of the person and the cup during the demonstration and during the automatic phase by the *environmental perception module v2* [4]

Obj.	**X (m)**	**Y (m)**	**Z (m)**
	During Demonstration		
Person	0.425	-0.773	0.372
Cup	0.611	0.059	0.050
	During Working Phase – Trial 1		
Person	0.291	-0.599	0.372
Cup	0.545	0.169	0.050
	During Working Phase – Trial 2		
Person	0.335	-0.733	0.372
Cup	0.505	0.249	0.050
	During Working Phase – Trial 3		
Person	0.335	-0.733	0.372
Cup	0.611	0.210	0.050
Box	0.603	-0.050	0.011

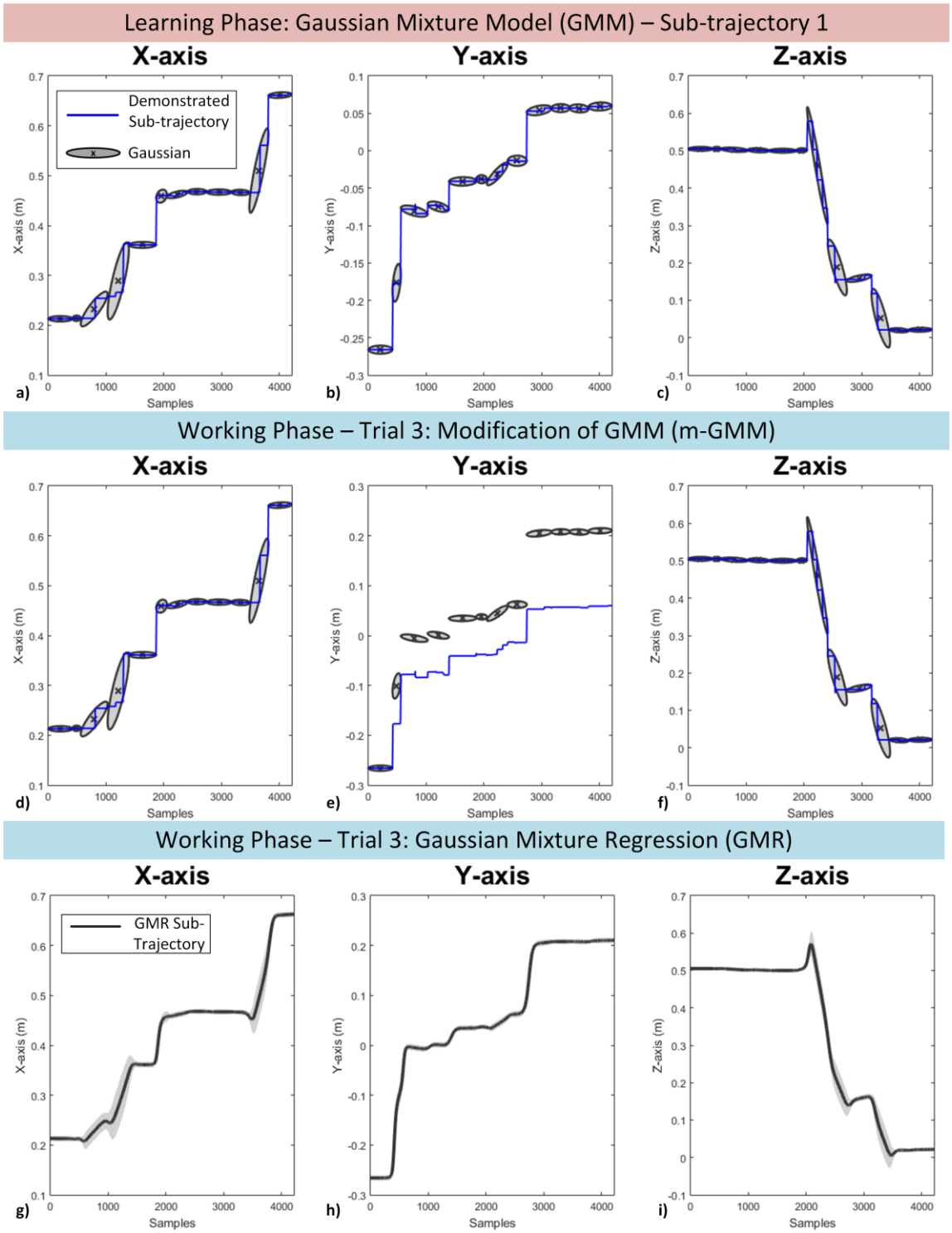


Figure 49: The learned GMM for the first sub-trajectory along the a) X-axis, b) Y-axis, c) Z-axis. The modification of the GMM for the first sub-trajectory – Trial 3 along the d) X-axis, e) Y-axis, f) Z-axis. The first GMR sub-trajectory produced by the m-GMM along the g) X-axis, h) Y-axis, i) Z-axis.

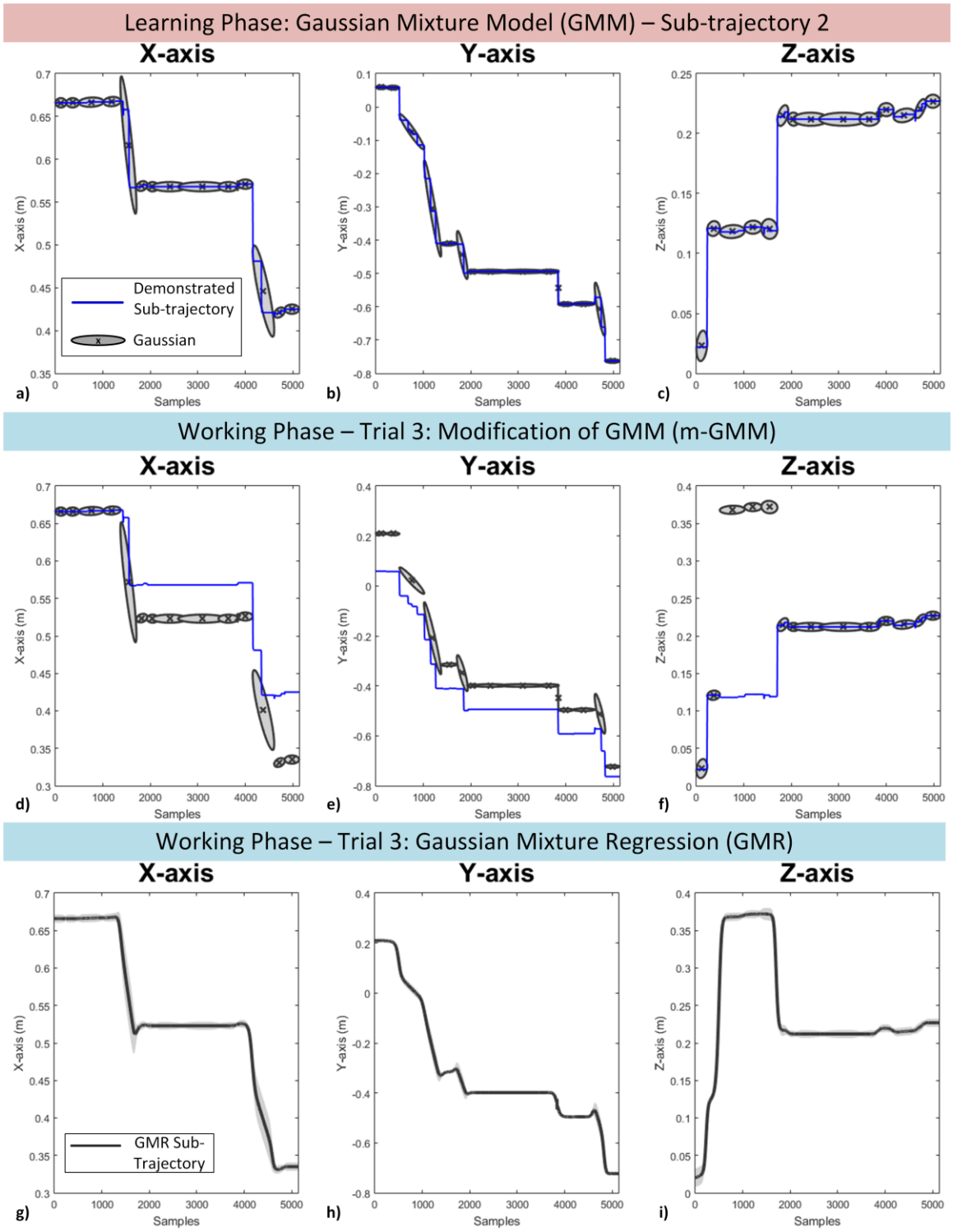


Figure 50: The learned GMM for the second sub-trajectory along the a) X-axis, b) Y-axis, c) Z-axis. The modification of the GMM for the second sub-trajectory – Trial 3 along the d) X-axis, e) Y-axis, f) Z-axis. The second GMR sub-trajectory produced by the m-GMM along the g) X-axis, h) Y-axis, i) Z-axis.

Firstly, the robot identified the task based on the objects in the scene. Subsequently, the robot confirmed that it was in the *Home* pose (which it was) and it suggested the next action, which was the *Close – Cup info*. In this scenario, the user confirmation had been disabled as only one sequence of actions was possible. The m-GMM/GMR algorithm was used to adapt the sub-trajectory to the new environmental conditions. In Figure 49d-f, the modification of the mean values of the learn GMM (output of m-GMM) is presented for the third trial in the working phase. The GMR method generated the adapted sub-trajectory to be followed by the robot, as shown in Figure 49g-i. After the robot executed the first sub-trajectory, the next action is suggested, which is the *End – Person info*. Similarly, the m-GMM and GMR algorithms were used to adapt the sub-trajectory to the new environmental conditions, as shown in Figure 50d-f and Figure 50g-i, respectively. The obstacle avoidance in m-GMM is illustrated in Figure 50f and it was calculated by adding, the height of the obstacle plus the height of the picked cup plus a safety distance to the mean values of the Gaussians in obstacle (as explained in Section 3.4.2.1). The m-GMM/GMR outputs for trial 1 and 2 are shown in appendix D.

The robot's end-effector trajectory for the demonstration during the learning phase and for all the three trials during the working phase is shown in Figure 51. In all three trials the robot completed the task successfully. The avoidance of the box (obstacle) during the third trial is marked in Figure 51.

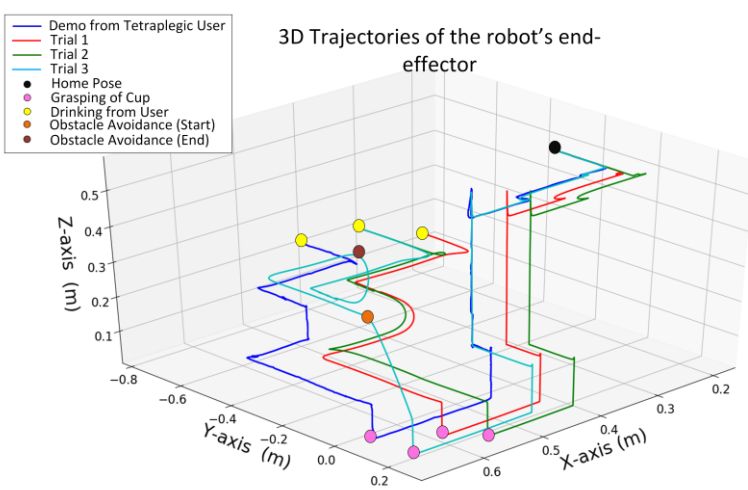


Figure 51: The trajectories of the end-effector in X, Y, Z-axis (3D) during the demonstration (demo) from the tetraplegic user in the learning phase and the three trials in the working phase [4]

5.5 Discussion

In this chapter the presented RLfD framework enables an assistive robotic manipulator to learn from a tetraplegic user to assist in drinking. The robot was able to learn the necessary moving actions and to reproduce the demonstrated task in changed environments. A head gesture-based HRI is used to enable a tetraplegic user to provide a demonstration for the assistive manipulation task.

The tetraplegic end-user was asked to fill in an additional questionnaire, the results of which are shown in Table 21. The questionnaire was used to evaluate the user acceptability of the presented system. As it can be observed, the feedback from the end-user was very positive. One important remark was that the tetraplegic end-user was in favor of using the 'hands-free' HRI to provide the demo herself.

However, the average time of the demonstration and the effort from the user are very high. The tetraplegic user needed 738sec (12.3 minutes) and 72 gestures to provide the demonstration, which is high. In future, head gestures could be combined with other modalities, such as speech to reduce the state machine for the robot control (presented in section 3.2.2.2). Additionally, the combination of 'hands-free' HRI with object recognition could assist the user to provide faster demonstration and with better quality.

Table 21: Questionnaire of the tetraplegic end-user

Questions	Possible Answers	Answer of the end-user
1. What is your first impression from the system?	Positive/Negative/ Scary/Strange	Positive
2. Can the system be supportive in your daily activities like drinking, eating?	Very Supportive/ Yes/A little/No	Very Supportive
3. Who would you prefer to control the robot during demonstration?	Myself/ Another person	Myself
4. Are you satisfied with the task learning and performing of the robot?	Very Satisfied/ Yes/A little/No	Very Satisfied

6. Conclusions and Outlook

6.1 Thesis Summary and Conclusions

The broad vision of this thesis is to enable robots to learn object manipulation tasks from human demonstrations and to be personalized for their users. This thesis has developed a robot learning framework that endows robots with the following abilities:

- learning the high-level sequence of actions from a demonstrated task, including the necessary moving actions (low-level),
- modifying the learned moving actions to accommodate different poses of involved objects and obstacles,
- performing a new task using the gain knowledge from previous learned tasks, and
- learning user's preferences regarding the sequence of actions for a task.

The development of the framework included methods for Human-Robot Interaction (HRI) and algorithms that enhance the field of Robot Learning from Demonstration (RLfD). The feasibility of this framework has been evaluated on assistive and industrial robotic tasks.

Chapter 1 discussed the motivation, the open topics in RLfD, and the main contributions of the thesis. Chapter 2 provided a theoretical background about machine learning techniques and discussed the state-of-the-art algorithms in RLfD. Chapter 3 described the developed RLfD framework and emphasized its main novelties. These are: it is able to learn sequence of high-level actions, including the moving actions (low-level)

from a demonstrated task; it aligns and selects similar demonstrated trajectories; it supports the user during the working phase by proposing sequence of actions and keeping the user in the loop; it utilizes prior knowledge to perform an unseen task consisting of previous learned tasks; it learns the user's preferences; it modifies the learned GMM to enable the robot to adapt to changes in the environment including obstacle avoidance (m-GMM method); it supports several HRI methods; it enables learning from single and multiple demonstrations.

Chapter 4 presented the results of the developed RLfD framework for two industrial manipulation tasks. The first task was a human-robot synergetic task to assemble a robot gripper. The robot was able to learn the sequence of actions, including the moving actions from multiple human demonstrations via kinesthetic teaching. The robot reproduced the task, even when the pose of the gripper parts were changed or an obstacle was present. The second task required the robot to insert pins into holes and a small study was conducted, where participants provided demonstrations via gamepad. The RLfD framework allowed the robot to use the prior knowledge to reproduce an unseen version of the task. Moreover, the robot learned the preferable sequence of high-level actions for two users. Furthermore, the 'hands-free' HRI concept was utilized to teach the robot additional moving actions needed to perform a new version of a known task. The results show the potential of the head gesture-based HRI to be used in future as a method to provide corrections in an already learned task. Both industrial tasks show the feasibility of the presented RLfD framework in industrial applications.

In Chapter 5, an application of the developed RLfD framework in assistive robotics was presented. The head gesture-based HRI enabled a person with tetraplegia to teach the robot by demonstrating an object manipulation task. The selected task was the manipulation of a cup, picking it from the table and bringing it close to their mouth. A small study with 13 participants (1 tetraplegic suffering from multiple sclerosis and 12 able-bodied) was conducted and all the participants were able to successfully provide demonstrations of the manipulation task. The results of the study show that a person with tetraplegia can effectively control assistive robotic manipulators. Furthermore, the presented RLfD framework enabled the robot to learn the sequence of actions from a

single demonstration, including the necessary moving actions for the object manipulation task. During the working phase, the robot successfully reproduced the learned task in different environmental conditions and the evaluation results were presented. The assistive manipulation task shows the potential of the RLfD framework to assist people with disabilities. Chapter 4 and 5 pointed out that the presented RLfD framework is generic, as it was implemented in two different robotic platforms.

6.2 Outlook

Although the results presented in this thesis show the feasibility of the developed RLfD, there are few open questions and future directions as a result of this thesis. These are summarized as follows.

The experiments in this thesis were performed in laboratory settings. The next logical step would be to perform experiments in industrial and home environments over extended time periods with end-users. Furthermore, larger sets of real-world tasks and strategies for long-term robot learning should be considered.

Additionally, robot vision (object recognition and pose estimation) is an important input to the robot learning framework. In this thesis, the robot vision was considered accurate. However, in the real world the robot vision is not perfect and it is a very challenging research topic. The presented RLfD framework would not succeed to manipulate an object, if the object is not accurately detected. The various HRI methods presented in this thesis could enable the user to control the robot whenever robot vision fails. For example, in an industrial synergetic task the user could perform some assembly actions and at the same time they could control the robot via a ‘hands-free’ HRI method to improve a picking position.

However, the ‘hands-free’ HRI methods are time consuming. The combination of multi-sensory HRI methods with robot vision may holds the key to enable users to provide ‘hands-free’ demonstrations faster and more comfortable. Moreover, personalization of HRI to enhance the user’s performance is an important social aspect, which should be improved in the future.

Abbreviations

ADL	Activity of Daily Living
ATTSA	Automatic Task Segmentation into Sequence of Actions
DG	Down-Gesture
DoF	Degrees of Freedom
DS	Decision Support
DSA	Decision Support and Adaptation
DTW	Dynamic Time Warping
GMM	Gaussian Mixture Model
GMR	Gaussian Mixture Regression
HRI	Human-Robot Interaction
IMU	Inertial Measurement Unit
IR	Infrared
IRL	Interactive Reinforcement Learning
ML	Machine Learning
LG	Left-Gesture
RG	Right-Gesture
RGB	Red-Green-Blue
RDP	Ramer-Douglas-Peucker
RL	Reinforcement Learning
RLfD	Robot Learning from Demonstration
SVM	Support Vector Machine
UG	Up-Gesture

Bibliography

Own publications

[1] M. Kyrarini, M. Haseeb, D. Ristić-Durrant and A. Gräser, "Robot Learning of Industrial Assembly Task via Human Demonstrations," *Autonomous Robots,* vol. 43, no. 1, pp. 239-257, 2019.

[2] M. Kyrarini, M. Haseeb, D. Ristic-Durrant and A. Gräser, "Robot Learning of Object Maminpulation Task Actions from Human Demonstrations," *Facta Universitatis Series: Mechanical Engineering (FU Mech Eng),* vol. 15, no. 2, pp. 217-229, 2017.

[3] M. Kyrarini, A. Leu, D. Ristic-Durrant, A. Gräser, A. Jackowski, M. Gebhard, J. Nelles, C. Bröhl, C. Brandl, A. Mertens and C. M. Schlick, "Human-Robot Sunergy for Cooperative Robots," *Facta Universitatis, Series: Automatic Control and Robotics,* vol. 15, no. 3, pp. 187-204, 2016.

[4] M. Kyrarini, Q. Zheng, M. Haseeb and A. Gräser, "Robot Learning of Assistive Manipulation Tasks by Demonstration via Head Gesture-based Interface," in *International Conference on Rehabilitation Robotics (ICORR 2019)*, Toronto, 2019.

[5] M. Haseeb, M. Kyrarini, S. Jiang, D. Ristic-Durrant and A. Gräser, "Head Gesture-based Control for Assistive Robots," in *11th ACM International Conference on PErvasive Technologies Related to Assistive Environments (PETRA)*, Corfu, 2018.

[6] M. Kyrarini and A. Gräser, "Selection of Human Demonstrations for Robot Learning of Industrial Scenario," in *2017 IEEE/RSJ IROS WORKSHOP Human in-the-loop robotic manipulation: on the influence of the human role*, 2017.

[7] M. Kyrarini, S. Naeem, X. Wang and A. Gräser, "Skill Robot Library: Intelligent Path Planning Framework for Object Manipulation," in *2017 25th European Signal Processing Conference (EUSIPCO)*, Kos, Greece, 2017.

References

[8] S. Nikolaidis and J. Shah, "Human-robot cross-training: Computational formulation, modeling and evaluation of a human team training strategy," in *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, Tokyo, 2013.

[9] H. Al Tair, T. Taha, M. Al-Qutayri and J. Dias, "Decentralized multi-agent POMDPs framework for humans-robots teamwork coordination in search and rescue," in *2015 International Conference on Information and Communication Technology Research (ICTRC)*, Abu Dhabi, 2015.

[10] R. Rocha, D. Portugal, M. Couceiro, F. Araujo, P. Menezes and J. Lobo, "The CHOPIN project: Cooperation between human and rObotic teams in catastrophic incidents," in *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, Linkoping, 2013.

[11] M. Daniele Comparetti, E. Beretta, M. Kunze, E. De Momi, J. Raczkowsky and G.

Ferrigno, "Event-based device-behavior switching in surgical human-robot interaction," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, 2014.

[12] Q. Zhao, D. Tu, S. Xu, H. Shao and Q. Meng, "Natural human-robot interaction for elderly and disabled healthcare application," in *2014 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, Belfast, 2014.

[13] A. Graser, T. Heyer, L. Fotoohi, U. Lange, H. Kampe, B. Enjarini, S. Heyer, C. Fragkopoulos and D. Ristic-Durrant, "A Supportive FRIEND at Work: Robotic Workplace Assistance for the Disabled," *IEEE Robotics & Automation Magazine,* vol. 20, no. 4, pp. 148 - 159, 2013.

[14] I. El Makrini, S. Elprama, J. Van den Bergh, B. Vanderborght, A. Knevels, C. Jewell, F. Stals, G. De Coppel, I. Ravyse, J. Potargent, J. Berte, B. Diericx, T. Waegeman and A. Jacobs, "Working with Walt: How a Cobot Was Developed and Inserted on an Auto Assembly Line," *IEEE Robotics & Automation Magazine ,* vol. 25, no. 2, pp. 51-58, 2018.

[15] "Neuartige Mensch-Roboter-Zusammenarbeit in der BMW Group Produktion," BMW Group, 10 September 2013. [Online]. Available: https://www.press.bmwgroup.com/deutschland/article/detail/T0209722DE/neuartige-mensch-roboter-zusammenarbeit-in-der-bmw-group-produktion?language=de.

[16] "KUKA Robotics," 2017. [Online]. Available: https://www.kuka.com/en-us/technologies/human-robot-collaboration. [Accessed 9 May 2017].

[17] "Meet the cobots: humans and robots together on the factory floor," Finance Times, 5 May 2016. [Online]. Available: https://www.ft.com/content/6d5d609e-02e2-11e6-af1d-c47326021344.

[18] "Porsche Employs Cobot on Assembly Line," Assembly , 18 December 2018. [Online]. Available: https://www.assemblymag.com/articles/94602-porsche-employs-cobot-on-assembly-line.

[19] A. Billard, S. Calinon, R. Dillmann and S. Schaal, "Chapter 59: Robot Programming by Demonstration," in *Springer Handbook of Robotics*, 2008.

[20] J. Zhang, Y. Wang and R. Xiong, "Industrial robot programming by demonstration," in *Int. Conf. on Advanced Robotics and Mechatronics (ICARM)*, 2016.

[21] B. D. Argall, S. Chernova, M. Veloso and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems,* vol. 57, no. 5, pp. 469-483, 2009.

[22] A. G. Billard, S. Calinon and R. Dillmann, "Learning from humans," in *Siciliano B., Khatib O. (eds) Springer Handbook of Robotics.*, Springer, Cham, 2016, pp. 1995-2014.

[23] S. Calinon, "Learning from Demonstration (Programming by Demonstration)," in *Encyclopedia of Robotics*, M. H. Ang, O. Khatib and B. Siciliano, Eds., Springer, 2019.

[24] Z. Zhu and H. Hu, "Robot Learning from Demonstration in Robotic Assembly: A Survey," *Robotics,* vol. 7, no. 2, p. 17, 2018.

[25] S. Calinon and D. Lee, "Learning control," *Humanoid Robotics: a Reference ,* pp. 1-52, 2016.

[26] R. Dillmann, T. Asfour, M. Do, R. Jäkel, A. Kasper, P. Azad, A. Ude, S. Schmidt-Rohr and M. Lösch, "Advances in robot programming by demonstration," *KI-Künstliche Intelligenz,* vol. 24, no. 4, pp. 295-303, 2010.

[27] P. Bakker and Y. Kuniyoshi, "Robot see, robot do: An overview of robot imitation.," in *AISB96 Workshop on Learning in Robots and Animals*, 1996.

[28] A. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," *IBM Journal of Research,* vol. 3, no. 3, pp. 535-554, 1957.

[29] T. Mitchell, Machine Learning, McGraw-Hill Science/Engineering/Math, 1997.

[30] A. Ng, "Machine Learning Yearning," 2018. [Online]. Available: https://www.mlyearning.org/.

[31] M. Awad and R. Khanna, Efficient learning machines: theories, concepts, and applications for engineers and system designers, Apress, 2015.

[32] J. Hurwitz and D. Kirsch, Machine Learning For Dummies, John Wiley & Sons, Inc, 2018.

[33] S. Shalev-Shwartz and S. Ben-David, Understanding machine learning: From theory to algorithms, Cambridge university press, 2014.

[34] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, Cambridge, MA: The MIT Press, 2018.

[35] J. Kober, J. A. Bagnell and J. Peters, "Reinforcement Learning in Robotics: A Survey," *International Journal of Robotics Research,* 2013.

[36] W. D. Smart and L. P. Kaelbling, "Effective reinforcement learning for mobile robots," in *IEEE International Conference on Robotics and Automation, 2002. Proceedings. ICRA'02.*, 2002.

[37] T. Lozano-Perez, "Robot programming.," *Proceedings of the IEEE,* vol. 71, no. 7, pp. 821-841, 1983.

[38] A. Segre and G. DeJong, "Explanation-based manipulator learning: Acquisition of planning ability through observation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 1985.

[39] P. Kormushev, S. Calinon and D. Caldwell, "Robot Motor Skill Coordination with EM-based Reinforcement Learning," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Taipei, 2010.

[40] J. Kober and J. Peters, "Learning Motor Primitives for Robotics," in *2009 IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, 2009.

[41] M. Forbes, R. P. N. Rao, L. Zettlemoyer and M. Cakmak, "Robot Programming by Demonstration with Situated Spatial Language Understanding," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, Washington, 2015.

[42] S. Alsharif, O. Kuzmicheva and A. Gräser, "Gaze Gesture-Based Human Robot Interface.," in *Technische Unterstützungssysteme, die die Menschen wirklich wollen*, 2016.

[43] S. Dziemian, W. W. Abbott and A. Aldo Faisal, "Gaze-based teleprosthetic enables intuitive continuous control of complex robot arm use: Writing & drawing," in *6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 2016.

[44] I. Pathirage, K. Khokar, E. Klay, R. Alqasemi and R. Dubey, "A Vision based P300 Brain Computer Interface for Grasping using a Wheelchair-Mounted Robotic Arm," in *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, Wollongong, Australia, 2013.

[45] B. Mindermann, "Untersuchung eines hybriden Brain-Computer Interfaces (BCIs) zur optimalen Auslegung als Mensch-Maschine-Schnittstelle," Doctoral dissertation, Universität Bremen, 2018.

[46] N. Rudigkeit, M. Gebhard and A. Gräser, "Towards a user-friendly AHRS-based human-machine interface for a semi-autonomous robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, Workshop on Assistive Robotics for Individuals with Disabilities: HRI Issues and Beyond*, 2014.

[47] A. B. Sylvain Calinon, "Incremental Learning of Gestures by Imitation in a Humanoid Robot," in *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, Arlington, VA, USA, 2007.

[48] K. J. K. Panagiotis K. Artemiadis, "EMG-based Teleoperation of a Robot Arm Using Low-Dimensional Representation," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, USA, 2007.

[49] R. Krug and D. Dimitrovz, “Representing movement primitives as implicit dynamical systems learned from multiple demonstrations,” in *2013 16th International Conference on Advanced Robotics (ICAR)*, Montevideo, 2013.

[50] K. Kukliński, K. Fischer, I. Marhenke, F. Kirstein, V. Maria, D. Sølvason, N. Krüger and T. Savarimuthu, "Teleoperation for learning by demonstration: Data glove versus object manipulation for intuitive robot control.," in *2014 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2014.

[51] J. Leitner, M. Luciw, A. Forster and J. Schmidhuber, “Teleoperation of a 7 DOF Humanoid Robot Arm Using Human Arm Accelerations and EMG Signals,” in *International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, Montreal, Canada, 2014.

[52] G. Du, P. Zhang, J. Mai and Z. Li, "Markerless Kinect-Based Hand Tracking for Robot Teleoperation," *International Journal of Advanced Robotic Systems,* vol. 9, no. 36, 2012.

[53] C. P. Quintero, R. T. Fomena, A. Shademan, O. Ramirez and M. Jagersand, "Interactive Teleoperation Interface for Semi-Autonomous Control of Robot Arms," in *2014 Canadian Conference on Computer and Robot Vision*, Montreal, 2014.

[54] S. Calinon, F. D'halluin, E. Sauser, D. Caldwell and A. Billard, "Learning and Reproduction of Gestures by Imitation," *IEEE Robotics & Automation Magazine,* vol. 17, no. 2, pp. 44 - 54, 2010.

[55] S. Ekvall and D. Kragic, "Learning Task Models from Multiple Human Demonstrations," in *15th IEEE Int. Symposium on Robot and Human Interactive Communication (ROMAN)*, 2006.

[56] M. Nicolescu and M. Mataric, "Natural methods for robot task learning: Instructive demonstrations, generalization and practice.," in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, 2003.

[57] M. Pardowitz, S. Knoop, R. Dillmann and R. D. Zollner, "Incremental learning of tasks from user demonstrations, past experiences, and vocal comments," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics),* vol. 37, no. 2, pp. 322-332, 2007.

[58] R. Cubek, W. Ertel and G. Palm, "High-level learning from demonstration with conceptual spaces and subspace clustering," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[59] M. Fox and D. Long, "PDDL2.1: An extension to pddl for expressing temporal planning domains," *Journal of artificial intelligence research,* vol. 20, pp. 61-124, 2003.

[60] K. Qian, J. Xu, G. Gao, F. Fang and X. Ma, "Learning Under-Specified Object Manipulations from Human Demonstrations," in *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2018.

[61] D. Moore and I. Essa, "Recognizing multitasked activities from video using stochastic context-free grammar," in *AAAI/IAAI*, 2002.

[62] Y. Yang, Y. Li, C. Fermüller and Y. Aloimonos, "Robot Learning Manipulation Action Plans by" Watching" Unconstrained Videos from the World Wide Web," in *AAAI*, 2015.

[63] Y. Yang, Y. Aloimonos, C. Fermuller and E. E. Aksoy, "Learning the semantics of manipulation action," in *arXiv preprint arXiv:1512.01525.*, 2015.

[64] G. E. Hovland, P. Sikka and B. J. McCarragher, "Skill acquisition from human demonstration using a hidden markov model," in *1996 IEEE International Conference on Robotics and Automation*, 1996.

[65] M. Patel, J. M. Valls, D. Kragic, C. Henrik Ek and G. Dissanayake, "Learning object, grasping and manipulation activities using hierarchical HMMs," *Autonomous Robots,* vol. 37, no. 3, p. 317–331, October 2014.

[66] K. Lu, S. Zhang, P. Stone and X. Chen, "Robot Representation and Reasoning with Knowledge from Reinforcement Learning," in *arXiv:1809.11074v3*, 2018.

[67] S. A. Raza, B. Johnston and M. A. Williams, "Reward from demonstration in interactive reinforcement learning," in *The Twenty-Ninth International Flairs Conference*, 2016.

[68] A. J. Ijspeert, J. Nakanishi and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *IEEE International Conference on Robotics and Automation*, 2002.

[69] S. Schaal, J. Peters, J. Nakanishi and A. Ijspeert, "Learning Movement Primitives," in *Robotics Research. The Eleventh International Symposium: With 303 Figures*, P.

Dario and R. Chatila, Eds., 2005, pp. 561-572.

[70] S. Schaal, "Dynamic movement primitives-a framework for motor control in humans and humanoid robotics," *Adaptive motion of animals and machines,* pp. 261-280, 2006.

[71] D.-H. Park, H. Hoffmann, P. Pastor and S. Schaal, "Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields," in *8th IEEE-RAS Int. Conf. on Humanoid Robots*, 2008.

[72] P. Pastor, H. Hoffmann, T. Asfour and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *IEEE Int. Conf. on Robotics and Automation (ICRA)* , 2009.

[73] K. Mülling, J. Kober, O. Kroemer and J. Peters, "Learning to select and generalize striking movements in robot table tennis," *The International Journal of Robotics Research,* vol. 32, no. 3, pp. 263-279, 2013.

[74] P. Kormushev, S. Calinon and D. Caldwell, "Robot motor skill coordination with EM-based reinforcement learning," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[75] M. Deng, Y. Hu and Z. Li, "Reinforcement learning of dual-arm cooperation for a mobile manipulator with sequences of dynamical movement primitives," in *2017 2nd International Conference on Advanced Robotics and Mechatronics (ICARM)*, 2017.

[76] X. Yin and Q. Chen, "Learning nonlinear dynamical system for movement primitives," in *IEEE Int. Conf. on Systems, Man and Cybernetics (SMC)*, 2014.

[77] C. Chen, C. Yang, C. Zeng, N. Wang and Z. Li, "Robot learning from multiple demonstrations with dynamic movement primitive," in *2017 2nd International Conference on Advanced Robotics and Mechatronics (ICARM)*, 2017.

[78] S. M. Khansari-Zadeh and A. Billard, "Learning Stable Non-Linear Dynamical Systems with Gaussian Mixture Models," *IEEE Transaction on Robotics,* vol. 27, no. 5, pp. 943-957, 2011.

[79] M. Brand and A. Hertzmann, "Style machines," in Proceedings of the 27th annual conference on Computer graphics and interactive techniques, 2000.

[80] S. Calinon, F. Guenter and A. Billard, "Goal-directed imitation in a humanoid robot," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005.

[81] A. G. Billard, S. Calinon and F. Guenter, "Discriminative and adaptive imitation in uni-manual and bi-manual tasks," *Robotics and Autonomous Systems,* vol. 54, no. 5, pp. 370-384, 2006.

[82] S. Calinon, E. Sauser, A. Billard and D. Caldwell, "Evaluation of a probabilistic approach to learn and reproduce gestures by imitation," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2010.

[83] E. &. C. S. Pignat, "Learning adaptive dressing assistance from human demonstration," *Robotics and Autonomous Systems,* vol. 93, pp. 61-75, 2017.

[84] S. Calinon, F. Guenter and A. Billard, "On Learning, Representing and Generalizing a Task in a Humanoid Robot," IEEE Transactions on Systems, Man

and Cybernetics, Part B, Special issue on robot learning by observation, demonstration and imitation, vol. 37, no. 2, pp. 286-298, 2007.

[85] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," *Intelligent Service Robotics,* vol. 9, no. 1, pp. 1-29, 2016.

[86] S. Calinon, " Robot learning with task-parameterized generative models," *Robotics Research ,* pp. 111-126, 2018.

[87] L. Rozo, S. Calinon, D. G. Caldwell, P. Jimenez and C. Torras, "Learning physical collaborative robot behaviors from human demonstrations," *IEEE Transactions on Robotics,* vol. 32, no. 3, pp. 513-527, 2016.

[88] M. Večerík, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe and M. Riedmiller, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," in *arXiv preprint arXiv:1707.08817.*, 2017.

[89] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband and G. Dulac-Arnold, "Deep q-learning from demonstrations," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[90] S. Calinon and A. Billard, "Incremental Learning of Gestures by Imitation in a Humanoid Robot," in *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, Arlington, VA, USA, 2007.

[91] H. Dindo and G. Schillaci, "An adaptive probabilistic approach to goal-level imitation learning," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[92] B. Akgun and A. Thomaz, "Simultaneously learning actions and goals from demonstration," *Autonomous Robots,* vol. 40, no. 2, pp. 211-227, 2016.

[93] G. Canal, E. Pignat, G. Alenyà, S. Calinon and C. Torras, "Joining high-level symbolic planning with low-level motion primitives in adaptive HRI: application to dressing assistance," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[94] Z. Yang, K. Merrick, L. Jin and H. A. Abbass, "Hierarchical Deep Reinforcement Learning for Continuous Action Control," *IEEE transactions on neural networks and learning systems,* vol. 99, pp. 1-11, 2018.

[95] D. Ding, H. Ka, C. Chung and H. Wang, "Shared Control of Assistive Robotic Manipulators," in *IEEE International Workshop Intelligent Robots and Systems (IROS)*, 2014.

[96] B. Argall, "Robot Learning from Motor-Impaired Teachers and Task Partners," 15 February 2017. [Online]. Available: https://simons.berkeley.edu/talks/brenna-argall-02-15-2017.

[97] B. D. Argall, "Machine Learning for Shared Control with Assistive Machines," in Proceedings of ICRA Workshop on Autonomous Learning: From Machine Learning to Learning in Real world Autonomous Systems, Karlsruhe, Germany, 2013.

[98] A. Goil, M. Derry and B. D. Argall, "Using machine learning to blend human and robot controls for assisted wheelchair navigation," in *2013 IEEE International Conference on Rehabilitation Robotics (ICORR)*, 2013.

[99] A. Broad, J. Arkin, N. Ratliff, T. Howard and B. Argall, "Real-time natural language corrections for assistive robotic manipulators," *The International Journal of Robotics Research,* vol. 36, no. 5-7, pp. 684-698, 2017.

[100] D. Kuhner, L. Fiederer, J. Aldinger, F. Burget, M. Völker, R. Schirrmeister, C. Do, J. Boedecker, B. Nebel, T. Ball and W. Burgard, "Deep Learning Based BCI Control of a Robotic Service Assistant Using Intelligent Goal Formulation," in *bioRxiv, 282848*, 2018.

[101] D. H. Grollman and A. G. & Billard, "Robot learning from failed demonstrations," *International Journal of Social Robotics,* vol. 4, no. 4, pp. 331-342, 2012.

[102] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler and A. Ng, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, 2009.

[103] Q. Fang, "Reliable 3D point cloud based object recognition," Master thesis, University of Bremen, 2016.

[104] A. Yassin, "Object recognition for robot manipulation," Master Thesis, University of Bremen, 2018.

[105] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," in *arXiv preprint arXiv:1804.02767*, 2018.

[106] M. A. Haseeb, J. Guan, D. Ristić-Durrant and A. Gräser, "DisNet: A Novel Method for Distance Estimation from Monocular Camera," in *10th Planning, Perception and Navigation for Intelligent Vehicles, IROS*, 2018.

[107] M. Haseeb, D. Ristic-Durrant and A. Gräser, "Long-range Obstacle Detection from a Monocular Camera," in *ACM Computer Science in Cars Symbosium, ECCV*, 2018.

[108] V. Arakelian, "Gravity compensation in robotics," *Advanced Robotics ,* vol. 30, no. 2, pp. 79-96, 2016.

[109] "F710 WIRELESS GAMEPAD," Logitech, [Online]. Available: https://www.logitechg.com/en-us/product/f710-wireless-gamepad. [Accessed 7 December 2018].

[110] R. O. Duda, P. E. Hart and D. G. Stork, Pattern classification, John Wiley & Sons, 2012.

[111] S. Mitra and T. Acharya, "Gesture recognition: A survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews),* vol. 37, no. 3, pp. 311-324, 2007.

[112] J. Huang, X. Shao and H. Wechsler, "Face pose discrimination using support vector machines (SVM)," in *Proceedings. Fourteenth International Conference on Pattern Recognition (Cat. No. 98EX170)*, 1998.

[113] L. P. Morency, C. Sidner, C. Lee and T. Darrell, "Contextual recognition of head gestures," in *Proceedings of the 7th international conference on Multimodal interfaces*, 2005.

[114] A. Kapoor and R. W. Picard, "A real-time head nod and shake detector," in *Proceedings of the 2001 workshop on Perceptive user interfaces*, 2001.

[115] P. C. Ng and L. C. De Silva, "Head gestures recognition," in Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205), 2001.

[116] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their applications,* vol. 13, no. 4, pp. 18-28, 1998.

[117] "myAHRS+, 3 Dimensional Attitude and Heading Reference System," WITHROBOT, [Online]. Available: http://withrobot.com/en/sensor/myahrsplus/. [Accessed 5 December 2018].

[118] "Nanny Web Cam Surveillance Camera Lxmimi Portable HD 1080P Mini Camera," LXMIMI CAM, [Online]. Available: https://www.amazon.de/gp/product/B07BF94CBY/ref=oh_aui_detailpage_o09_s00?ie=UTF8&psc=1. [Accessed 7 December 2018].

[119] J. Shi and C. Tomasi, "Good features to track," Cornell University, 1993.

[120] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence,* vol. 17, no. 1-3, pp. 185-203, 1981.

[121] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the 7th International Joint Conference on Artificial Intelligence – Vol. 2*, 1981.

[122] S. Schaal, "Learning from demonstration," in *Advances in Neural Information Processing Systems 9*, 1997.

[123] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," *KDD workshop,* vol. 10, no. 16, pp. 359-370, 1994.

[124] B. Akgun, M. Cakmak, K. Jiang and A. Thomaz, "Keyframe-based Learning from Demonstration," *Int. J. of Social Robotics,* vol. 4, no. 4, pp. 345-355, 2012.

[125] E. Sabbaghi, M. Bahrami and S. Ghidary, "Learning of gestures by imitation using a monocular vision system on a humanoid robot," in *Second RSI/ISM Int. Conf. on Robotics and Mechatronics (ICRoM)*, 2014.

[126] A. Movchan and M. Zymbler, "Time series subsequence similarity search under dynamic time warping distance on the intel many-core accelerators," in *Int.Conf. on Similarity Search and Applications*, 2015.

[127] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Computer Graphics and Image Processing,* vol. 1, no. 3, pp. 244-256, 1972.

[128] D. Douglas and T. Peucker, "Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature," *Cartographica The International Journal for Geographic Information and Geovisualization,* vol. 10, no. 2, pp. 112-122, 1973.

[129] R. Zacharski, "A Programmer's Guide to Data Mining," 2015. [Online]. Available: http://guidetodatamining.com/.

[130] S. Calinon, Robot programming by demonstration: A probabilistic approach, EPFL Press, 2009.

[131] A. Dempster, N. Laird and D. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society, Series B.,* vol. 39, no. 1, pp. 1-38, 1977.

[132] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability,* vol. 1, no. 14, pp. 281-297, 1967.

[133] G. Schwarz, "Estimating the dimension of a model," *The annals of statistics,* vol. 6, no. 2, pp. 461-464, 1978.

[134] C. Hennig, "Methods for merging Gaussian mixture components," *Advances in data analysis and classification,* vol. 4, no. 1, pp. 3-34, 2010.

[135] F. J. Fabozzi, S. M. Focardi, S. T. Rachev and B. G. Arshanapalli, "Model Selection Criterion:AIC and BIC," in *The basics of financial econometrics: Tools, concepts, and asset management applications*, John Wiley & Sons, Inc, 2014, pp. 399-403.

[136] D. Hershberger, D. Gossow and J. Faust, "3D visualization tool for ROS.," ROS.org, 16 May 2018. [Online]. Available: http://wiki.ros.org/rviz.

[137] F. Cruz, S. Magg, C. Weber and S. Wermter, "Training agents with interactive reinforcement learning and contextual affordances," *IEEE Transactions on Cognitive and Developmental Systems,* vol. 8, no. 4, pp. 271-284, 2016.

[138] C. Watkins and P. Dayan, "Q-learning," *Machine learning,* vol. 8, no. 3-4, pp. 279-292, 1992.

[139] F. Cruz, S. Magg, C. Weber and S. Wermter, "Training agents with interactive reinforcement learning and contextual affordances," *IEEE Transactions on Cognitive and Developmental Systems,* vol. 8, no. 4, pp. 271-284, 2016.

[140] Y. W. Seo and B. T. Zhang, "A reinforcement learning agent for personalized information filtering," in *Proceedings of the 5th international conference on Intelligent user interfaces*, 2000.

[141] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg and D. Amodei, "Deep reinforcement learning from human preferences.," in *Advances in Neural Information Processing Systems*, 2017.

[142] R. Akrour, M. Schoenauer and M. Sebag, "April: Active preference learning-based reinforcement learning.," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2012.

[143] S. Zaidenberg and P. Reignier, "Reinforcement Learning of User Preferences for a Ubiquitous Personal Assistant," in *Advances in Reinforcement Learning*, 2011.

[144] "Workerbot3," pi4, [Online]. Available: https://www.pi4.de/fileadmin/material/datenblatt/Datenblatt_WB3_EN_V1_2.pdf. [Accessed 5 March 2019].

[145] "Universal Robot UR10," Universal Robots, [Online]. Available: https://www.universal-robots.com/products/ur10-robot/. [Accessed 5 March 2019].

[146] "Kinect for Xbox One (Wikipedia)," Microsoft, [Online]. Available: https://en.wikipedia.org/wiki/Kinect#Kinect_for_Xbox_One_(2013). [Accessed 5 March 2019].

[147] "KINOVA Ultra Lightweight robotic arm 7 DOF Spherical - Specifications," [Online]. Available: https://www.kinovarobotics.com/sites/default/files/ULWS-RA-JAC-7DS-SP-INT-EN%20201804-1.2%20%28KINOVA%E2%84%A2%20Ultra%20lightweight%20robotic%20arm%207%20DOF%20Spherical%20Specifications%29_0.pdf. [Accessed 26 February 2019].

[148] V. Maheu, P. S. Archambault, J. Frappier and F. Routhier, "Evaluation of the JACO robotic arm: Clinico-economic study for powered wheelchair users with upper-extremity disabilities," in *2011 IEEE International Conference on Rehabilitation Robotics*, 2011.

[149] F. Routhier, P. Archambault, M. Cyr, V. Maheu, M. Lemay and I. Gélinas, " Benefits of JACO robotic arm on independent living and social participation: an exploratory study," in *RESNA Annual Conference*, 2014.

[150] "Intel® RealSense™ Depth Camera D435," Intel, [Online]. Available: https://click.intel.com/intelr-realsensetm-depth-camera-d435.html. [Accessed 29 January 2019].

[151] M. Carfagni, R. Furferi, L. Governi, C. Santarelli, M. Servi, F. Uccheddu and Y. Volpe, "Metrological and Critical Characterization of the Intel D415 Stereo Depth Camera," *Sensors,* vol. 19, no. 3, p. 489, 2019.

[152] C. S. Chung, H. Wang and R. A. Cooper, "Functional assessment and performance evaluation for assistive robotic manipulators: Literature review," *The journal of spinal cord medicine,* vol. 36, no. 4, pp. 273-289, 2013.

[153] V. Preedy and R. Watson, "5-Point Likert Scale," in *Handbook of Disease Burdens and Quality of Life Measures*, New York, NY, Springer, 2010.

[154] N. Rudigkeit, "AMiCUS-Bewegungssensor-basiertes Human-Robot Interface zur intuitiven Echtzeit-Steuerung eines Roboterarmes mit Kopfbewegungen: AMiCUS-Motion Sensor-based Human-Robot Interface for Intuitive Realtime Control of a Robot Arm Using Head Motion," Doctoral dissertation, Universität Bremen, 2017.

[155] "Douglas-Peucker N," psimpl - generic n-dimensional polyline simplification, [Online]. Available: http://psimpl.sourceforge.net/douglas-peucker.html. [Accessed 20 March 2017].

[156] D. R. Wright, "Finite state machines," Carolina State University, 2005.

Appendix A: Publications by the author

This appendix lists the publications by the author, which have been produced over the course of this thesis, and they are the basis of this thesis.

Related Journal Publications

1. **Kyrarini M.**, Haseeb M.A., Ristić-Durrant D., Gräser A., 2019. Robot Learning of Industrial Assembly Task via Human Demonstrations. *Autonomous Robots*, 43(1), pp. 239-257. Doi: 10.1007/s10514-018-9725-6
2. **Kyrarini M.**, Haseeb M.A., Ristić-Durrant D., Gräser A., 2017. Robot Learning of Object Manipulation Task Actions from Human Demonstrations. *Facta Universitatis, series: Mechanical Engineering (FU Mech Eng)*, 15(2), pp. 217-229. Doi: 10.22190/FUME170515010K
3. **Kyrarini M.**, Leu A., Ristić-Durrant D., Gräser A., Jackowski A., Gebhard M., Nelles J., Bröhl C., Brandl C., Mertens A., Schlick C. M., 2016. Human-Robot Synergy for cooperative robots. *Facta Universitatis, series: Autonomic Control and Robotics*, 15(3), pp. 187-204. Doi: 10.22190/FUACR1603187K

Related Conference Publications

1. **Kyrarini M.**, Zheng Q., Haseeb M.A, Gräser A., 2019. Robot Learning of Assistive Manipulation Tasks by Demonstration via Head Gesture-based Interface. In *International Conference on Rehabilitation Robotics (ICORR 2019)*, Toronto, Canada.
2. Haseeb M.A, **Kyrarini M.**, Jiang S., Ristić-Durrant D., Gräser A., 2018. Head Gesture-based Control for Assistive Robots. In *11th ACM International Conference on Pervasive Technologies Related to Assistive Environments (PETRA)*, Corfu Greece, pp. 379-383. ACM. doi: 10.1145/3197768.3201574
3. **Kyrarini M.**, Gräser A., 2017. Selection of Human Demonstrations for Robot Learning of Industrial Scenario. In *2017 IEEE/RSJ IROS WORKSHOP Human in-the-loop robotic manipulation: on the influence of the human role,* Vancouver, Canada (2-pages extended abstract)

4. **Kyrarini M.**, Naeem S., Wang X., Gräser A., 2017. Skill Robot Library: Intelligent Path Planning Framework for Object Manipulation. In *25th European Signal Processing Conference (EUSIPCO 2017)*, Kos island, pp.2398 – 2402. IEEE, doi: 10.23919/EUSIPCO.2017.8081640
5. **Kyrarini M.**, Gräser A., 2017. Learning robot manipulation tasks from human demonstrations. *51. Regelungstechnisches Kolloquium in Boppard*, Boppard, Germany (2-pages extended abstract)

Other Journal and Conference Publications

1. Goldau F.F., Shastha, T.K., **Kyrarini M.**, Gräser A., 2019. Autonomous Multi-Sensory Robotic Assistant for a Drinking Task. In *International Conference on Rehabilitation Robotics (ICORR 2019)*, Toronto, Canada.
2. Koller T.L., **Kyrarini M.**, Gräser A., 2019. Towards robotic drinking assistance: Low cost multi-sensor system to limit forces in Human-Robot-Interaction. In *12th ACM International Conference on PErvasive Technologies Related to Assistive Environments (PETRA)*, Rhodes Greece, pp. 243-246. ACM. doi: 10.1145/3316782.3321539
3. Tsiakas K., **Kyrarini M.**, Karkaletsis V., Makedon F., Korn O., 2018. A Taxonomy in Robot-Assisted Training: Current Trends, Needs and Challenges. *Technologies* 6, no. 4: 119. *MDPI.* Doi: 10.3390/technologies6040119
4. Fang Q., **Kyrarini M.**, Ristic-Durrant D., Gräser A., 2018. RGB-D Camera based 3D Human Mouth Detection and Tracking Towards Robotic Feeding Assistance. In *11th ACM International Conference on Pervasive Technologies Related to Assistive Environments (PETRA),* Corfu Greece, pp. 391-396. ACM. doi: 10.1145/3197768.3201576
5. Jiang S., Wang X., **Kyrarini M.**, Gräser A., 2017. A Robust Algorithm for Gait Cycle Segmentation. In *25th European Signal Processing Conference (EUSIPCO 2017),* Kos island, pp. 31 – 35. IEEE, doi: 10.23919/EUSIPCO.2017.8081163
6. Gao G., **Kyrarini M.**, Razavi M., Wang X., Gräser A., 2016. Comparison of Dynamic Vision Sensor-based and IMU-based Systems for Ankle Joint Angle Gait Analysis. In *2016 2nd International Conference on Frontiers of Signal Processing (ICFSP 2016),* pp. 93-98. IEEE. Doi: 10.1109/ICFSP.2016.7802963

7. Wang X., **Kyrarini M.**, Ristić-Durrant D., Spranger M., Gräser A., 2016. Monitoring of Gait Performance Using Dynamic Time Warping on IMU-Sensor Data. In *2016 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, pp. 1-6. IEEE. Doi: 10.1109/MeMeA.2016.7533745
8. **Kyrarini M.**, Wang X., Gräser A., 2015. Comparison of vision-based and sensor-based Systems for Joint Angle Gait Analysis. In *2015 IEEE International Symposium on Medical Measurements and Applications*, pp. 375-379, IEEE. Doi: 10.1109/MeMeA.2015.7145231
9. **Kyrarini M.**, Slavnić S., Ristić-Durrant D., 2014. Fuzzy controller for the control of the mobile platform of the CORBYS robotic gait rehabilitation system. *Facta Universitatis, series: Mechanical Engineering (FU Mech Eng)*, 12(3), pp. 223-234.

Appendix B: Polyline Simplification Ramer-Douglas-Peucker (RDP)

The Ramer-Douglas-Peucker (RDP) Algorithm is a polyline simplification, which is implemented as follows:

- initially, the first and the last point of the original trajectory creates a line and both points are added to simplified trajectory,
- the perpendicular distance is calculated between the created line and each intermediate point of the original trajectory,
- the intermediate point that has the maximum calculated distance from the created line and the calculated distance is larger than the tolerance specified by the user is added to the simplified points,
- the above process will recur for every point in the current simplified polyline until all the points of the original trajectory are within the specified tolerance.

In this thesis a variation of the original RDP algorithm is used, which is called Douglas-Peucker N [155]. The algorithm uses a point count tolerance. In other words, the algorithm requires as input the exact number of simplified points that are required. The points that have the maximum calculated distance from the created lines are the simplified points.

Appendix C: Experimental Results in Industrial Robotics Application

I. Robot Gripper Assembly Task (Task 1)

In this section, the complete results for the robot gripper assembly task (section 4.2) are presented. Table 22 associates the groups of sub-trajectories for the robot gripper assembly tasks and the figure number where the simplified sub-trajectories of each group are presented. In the figures, the *x*-,*y*-,*z*- dimensions are denoted as X-,Y-,Z-axis, respectively. The quaternions *qx, qy, qz, qw* are denoted as Quaternion-X,-Y,-Z,-W, respectively.

Table 22: Association between the groups of sub-trajectories and figures for the robot gripper assembly task

Group	Sub-trajectories between the high-level actions	Figure	Group	Sub-trajectories between the high-level actions	Figure
Left 1	*LV Home – Home* and *LV Close – Top part info*	Figure 52	Left 5	*LV Home – Home* and *LV Close – Left side part info*	Figure 55
Left 2	*LV Close – Top part info* and *LV Open – Base part info*	Figure 53	Left 6	*LV Close – Left side part info* and *LV Open – Base part info*	Figure 56
Left 3	*LV Open – Base part info* and *LV End – Table info*	Figure 54	Left 7	*LV Open – Base part info* and *LV End – Table info*	Figure 57
Right 1	*RV Home – Home* and *RV Close – Black part info*	Figure 25	Right 5	*RV Home – Home* and *RV Close – Right side part info*	Figure 58
Right 2	*RV Close – Black part info* and *RV Open – Base part info*	Figure 26	Right 6	*RV Close – Right side part info* and *RV Open – Base part info*	Figure 59
Right 3	*RV Open – Base part info* and *RV End – Table info*	Figure 27	Right 7	*RV Open – Base part info* and *RV End – Table info*	Figure 60

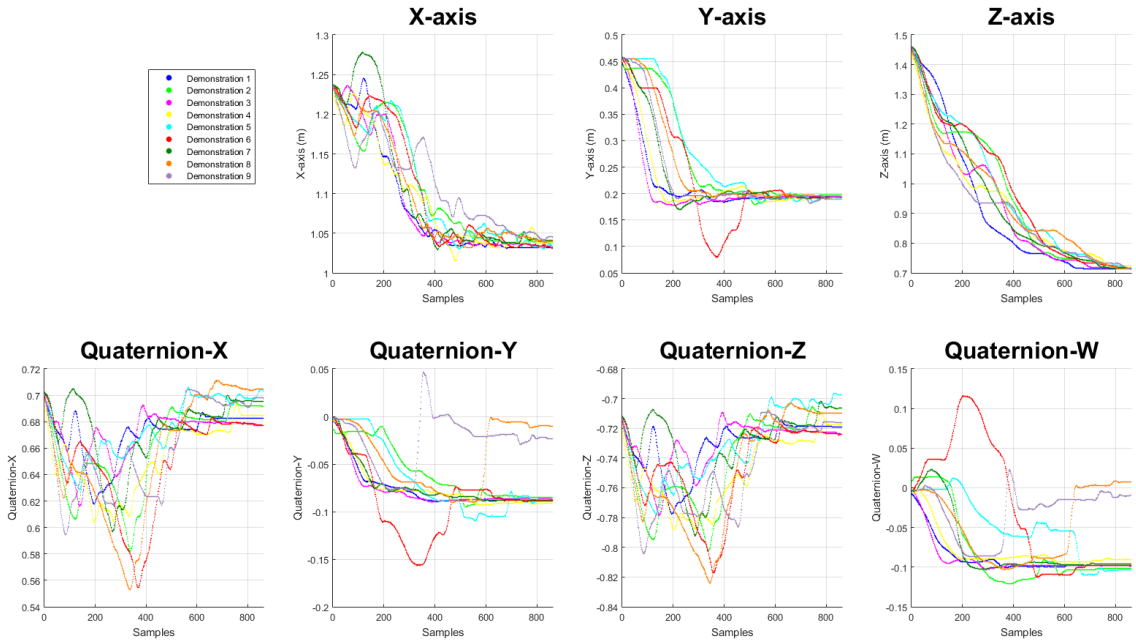


Figure 52: Robot gripper assembly task - The sub-trajectories of the Left 1 group after Ramer-Douglas-Peucker simplification

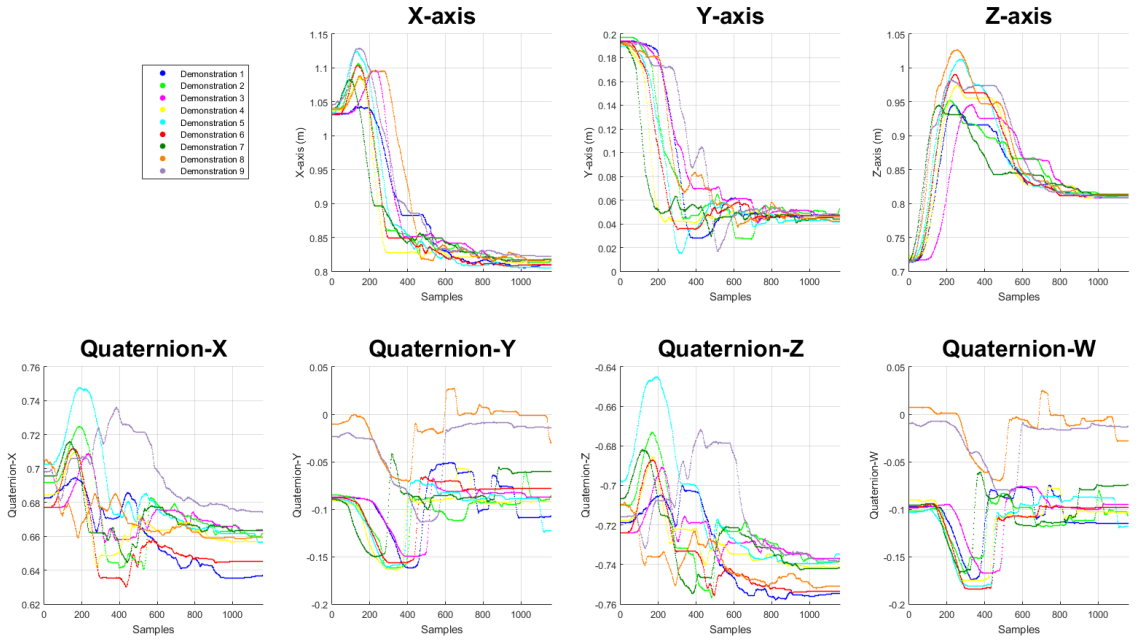


Figure 53: Robot gripper assembly task - The sub-trajectories of the Left 2 group after Ramer-Douglas-Peucker simplification

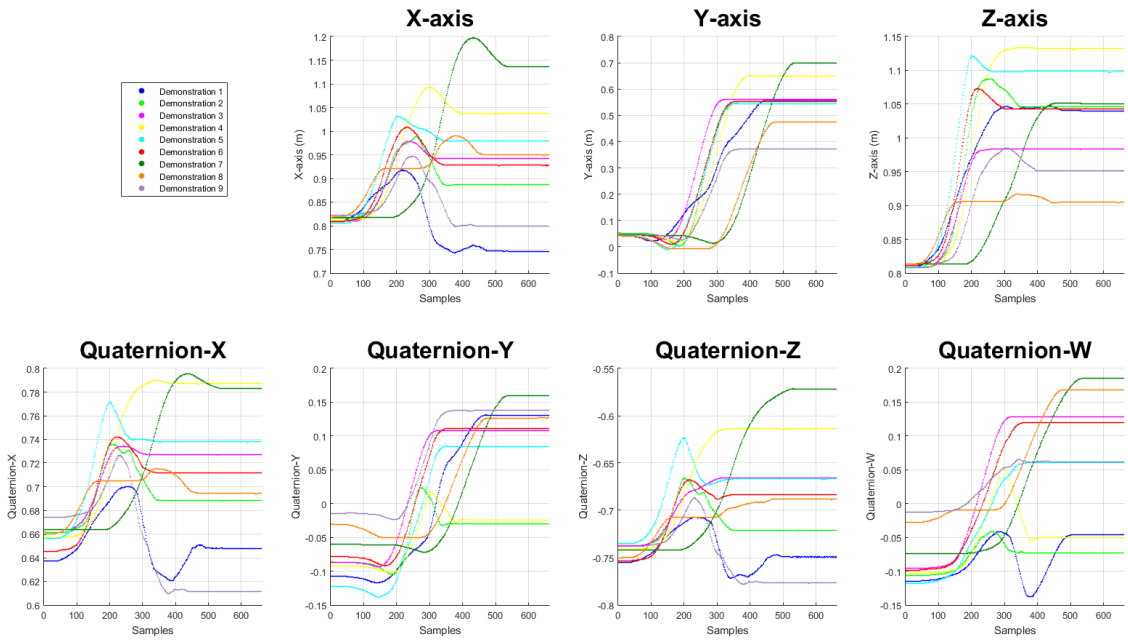


Figure 54: Robot gripper assembly task - The sub-trajectories of the Left 3 group after Ramer-Douglas-Peucker simplification

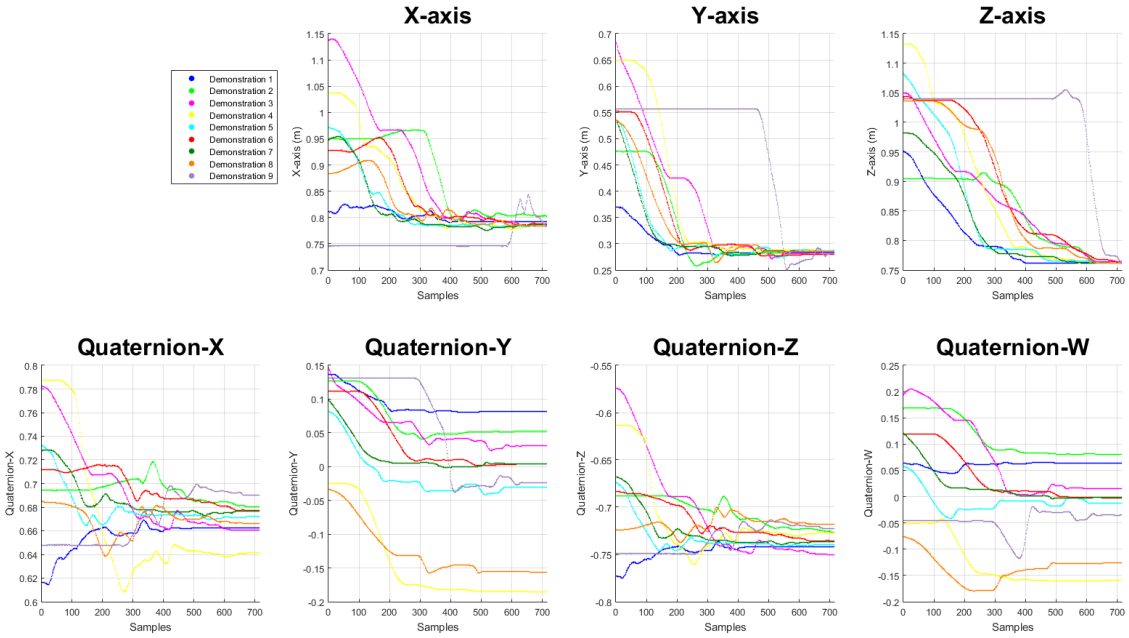


Figure 55: Robot gripper assembly task - The sub-trajectories of the Left 5 group after Ramer-Douglas-Peucker simplification

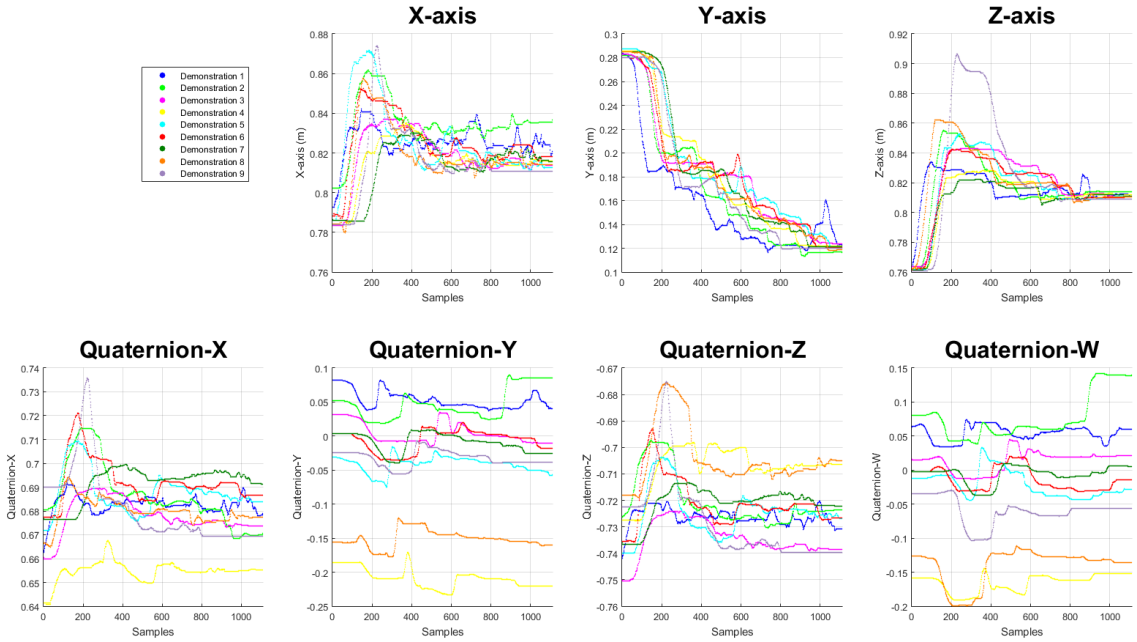


Figure 56: Robot gripper assembly task - The sub-trajectories of the Left 6 group after Ramer-Douglas-Peucker simplification

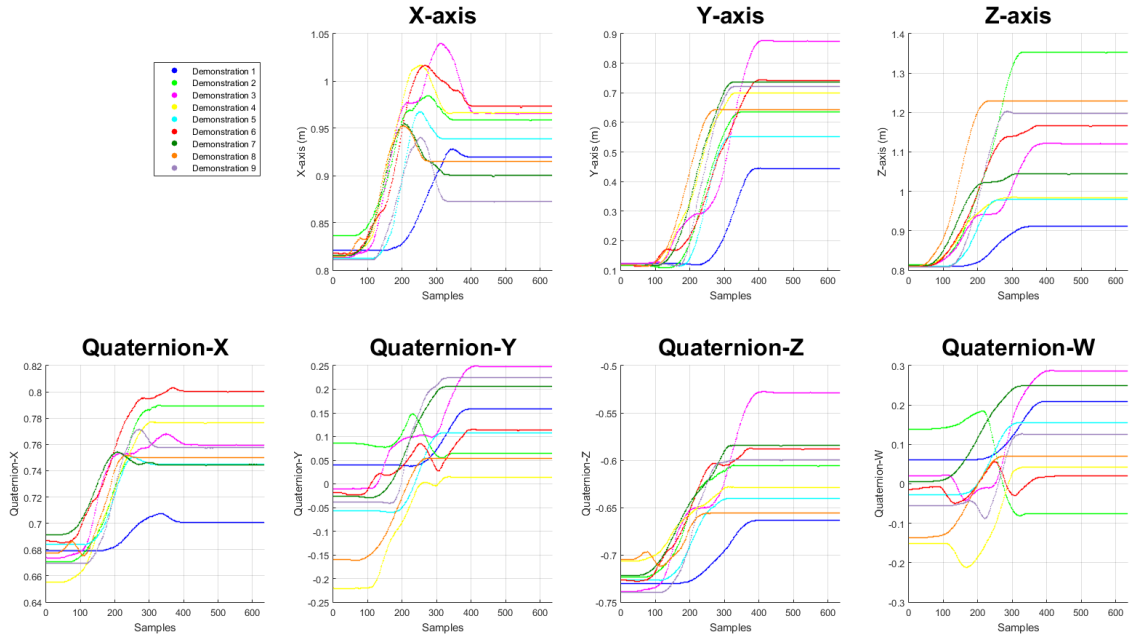


Figure 57: Robot gripper assembly task - The sub-trajectories of the Left 7 group after Ramer-Douglas-Peucker simplification

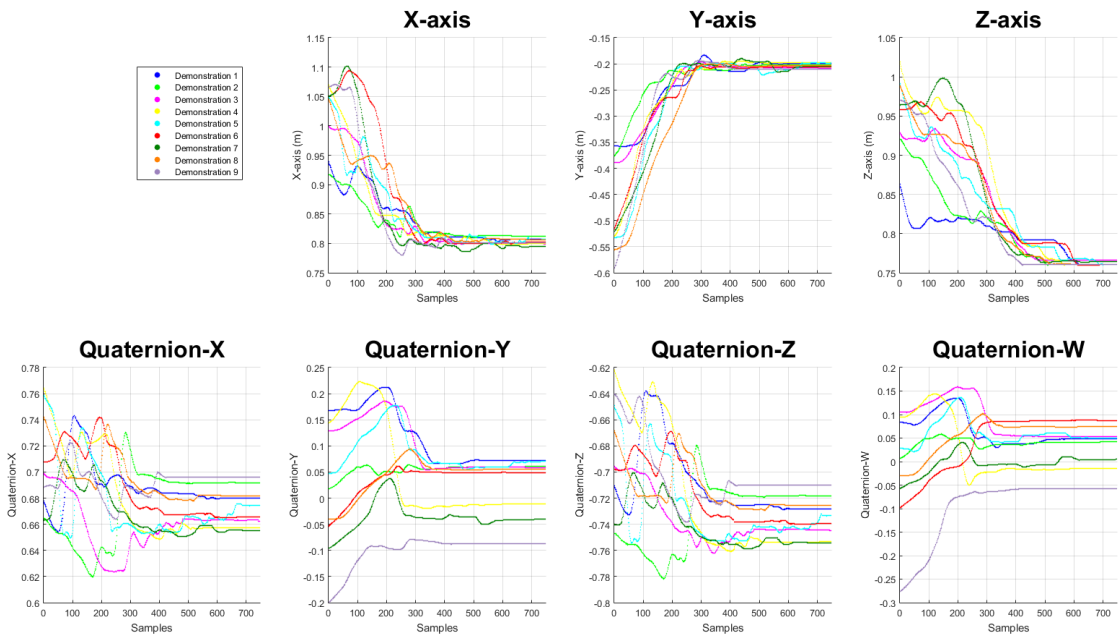


Figure 58: Robot gripper assembly task - The sub-trajectories of the Right 5 group after Ramer-Douglas-Peucker simplification

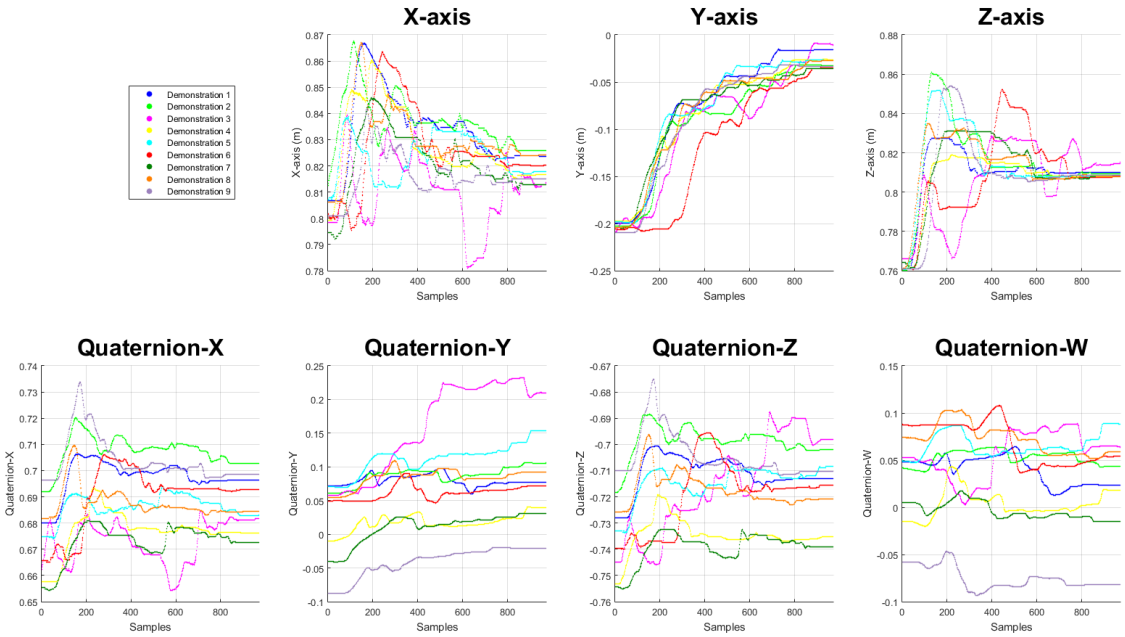


Figure 59: Robot gripper assembly task - The sub-trajectories of the Right 6 group after Ramer-Douglas-Peucker simplification

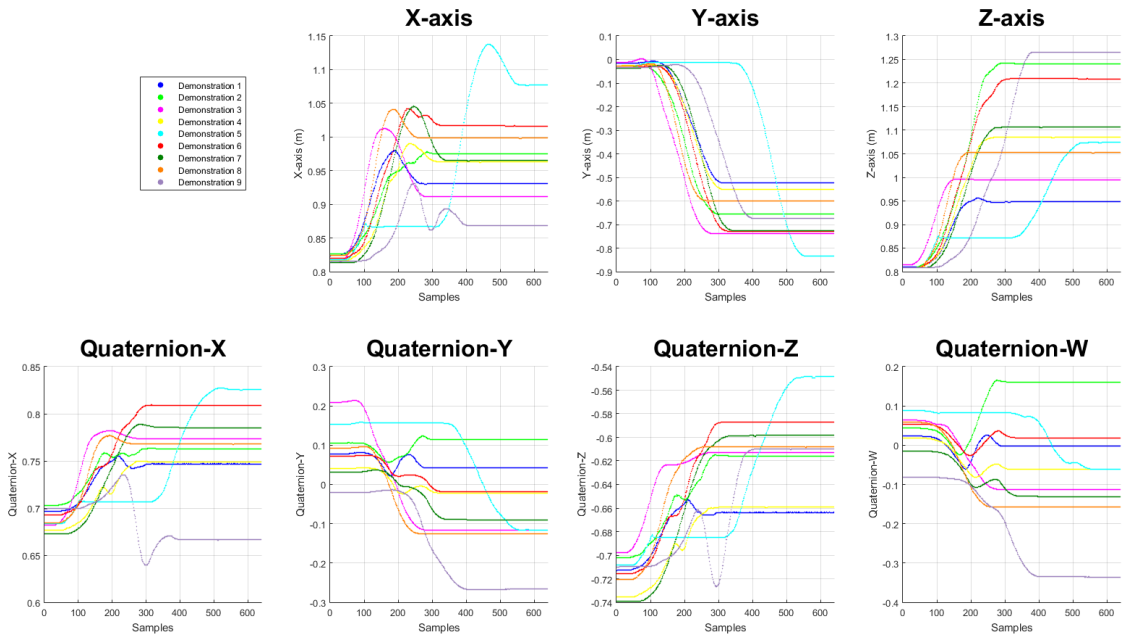


Figure 60: Robot gripper assembly task - The sub-trajectories of the Right 7 group after Ramer-Douglas-Peucker simplification

Moreover the learned GMM, the output m-GMM, and the GMR for the working phase are presented. Table 23 associates the groups for the robot gripper assembly tasks with the figure number, where the learned GMM, m-GMM and GMR are illustrated. In the figures, the *x*-, *y*-, *z*- dimensions are denoted as X-,Y-,Z-axis, respectively. The quaternions *qx, qy, qz, qw* are denoted as Quaternion-X,-Y,-Z,-W, respectively.

Table 23: Association between the groups of sub-trajectories and figures for the robot gripper assembly task

Group	**Figure GMM/m-GMM/GMR for *x*-, *y*-, *z*-dimensions**	**Figure GMM/m-GMM/GMR for quaternions**	**Group**	**Figure GMM/m-GMM/GMR for *x*-, *y*-, *z*-dimensions**	**Figure GMM/m-GMM/GMR for quaternions**
Left 1	Figure 61	Figure 62	Left 5	Figure 67	Figure 68
Left 2	Figure 63	Figure 64	Left 6	Figure 69	Figure 70
Left 3	Figure 65	Figure 66	Left 7	Figure 71	Figure 72
Right 1	Figure 28	Figure 29	Right 5	Figure 73	Figure 74
Right 2	Figure 30	Figure 31	Right 6	Figure 75	Figure 76
Right 3	Figure 32	Figure 33	Right 7	Figure 77	Figure 78

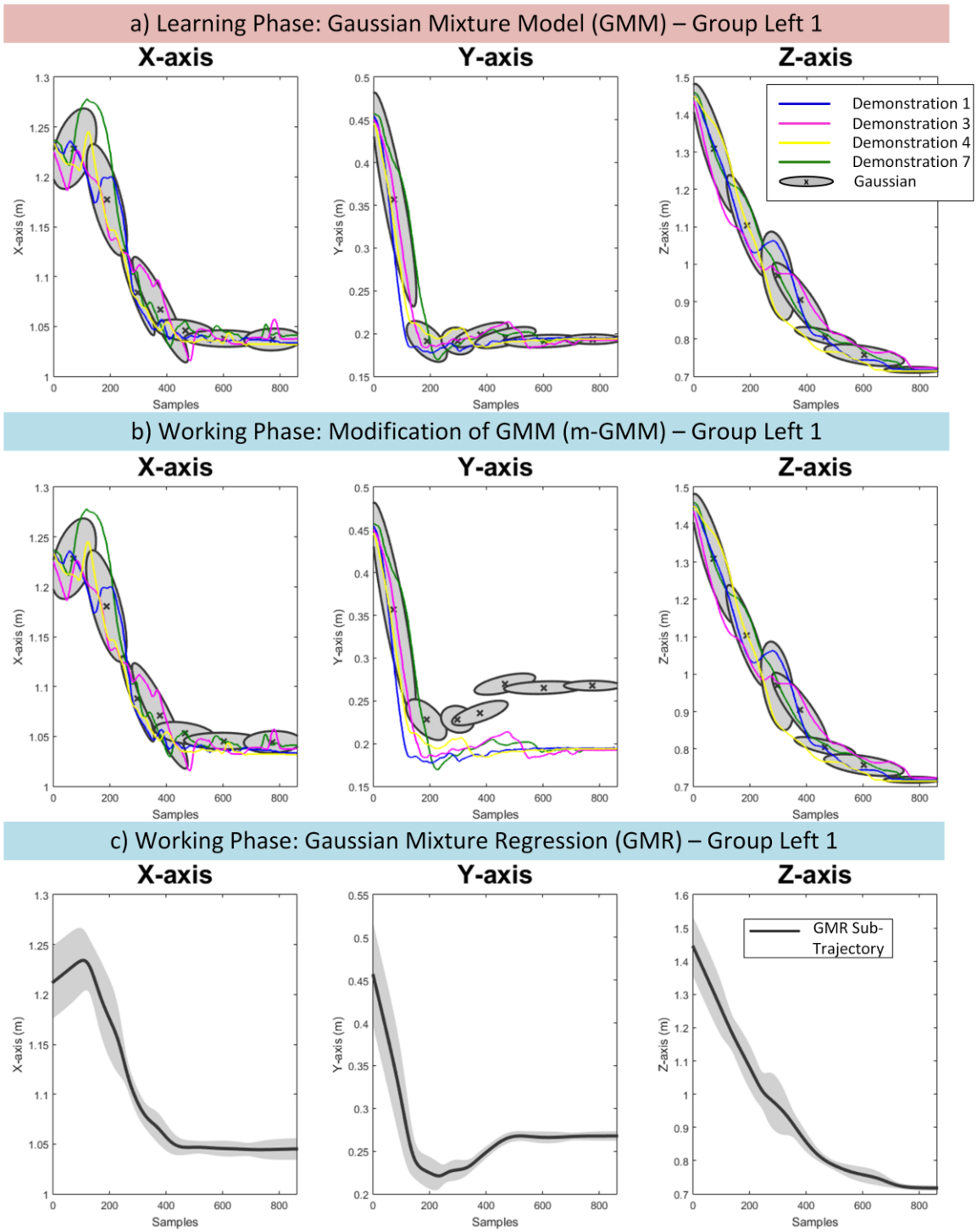


Figure 61: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 1 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 1 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 1 along the X-, Y-, and Z-axis.

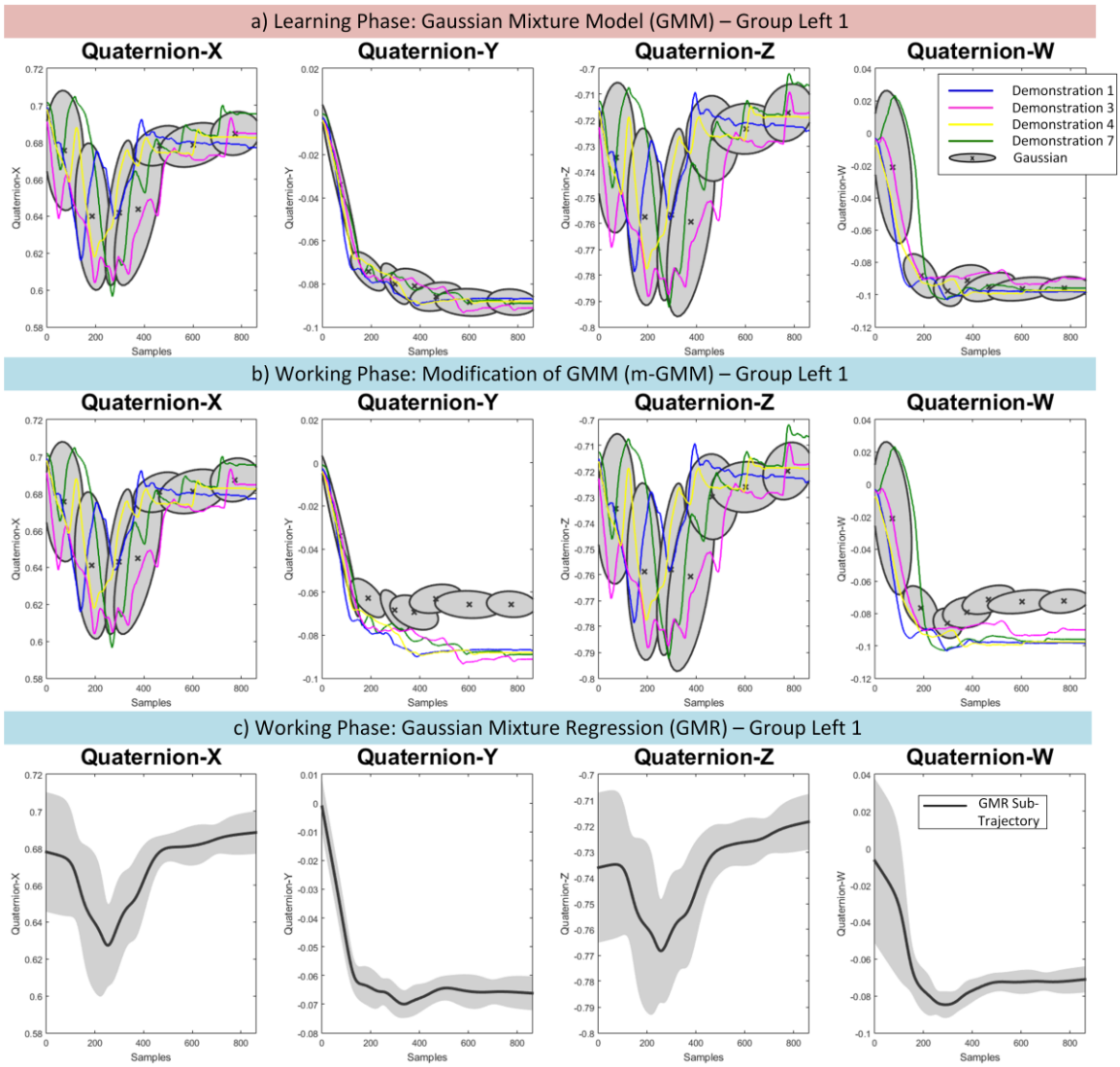


Figure 62: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 1 for the quaternions. b) The modification of the learned GMM for the sub-trajectories of group Left 1 during the trial 1 for the quaternions. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 1 for the quaternions.

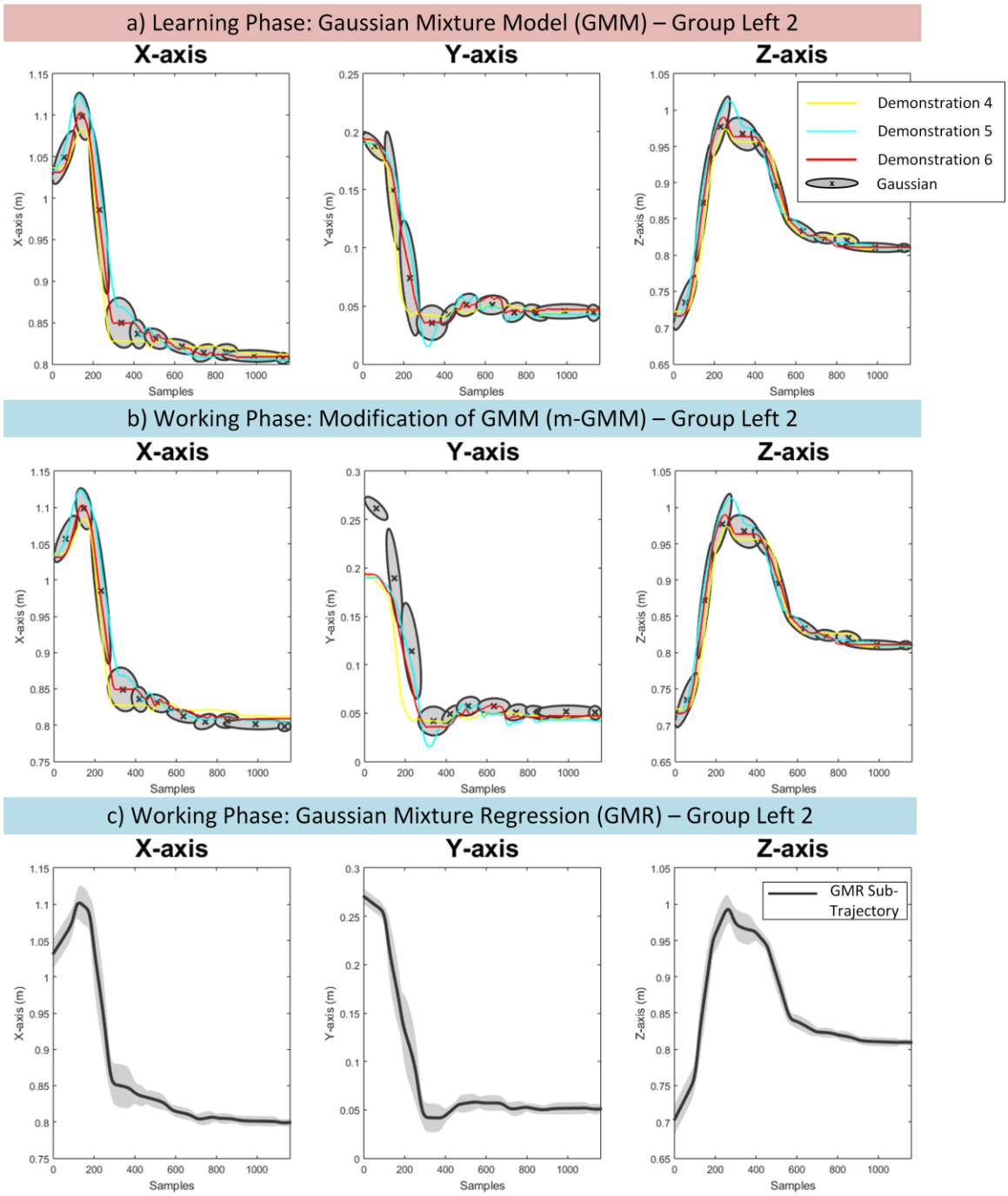


Figure 63: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 2 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 2 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 2 along the X-, Y-, and Z-axis.

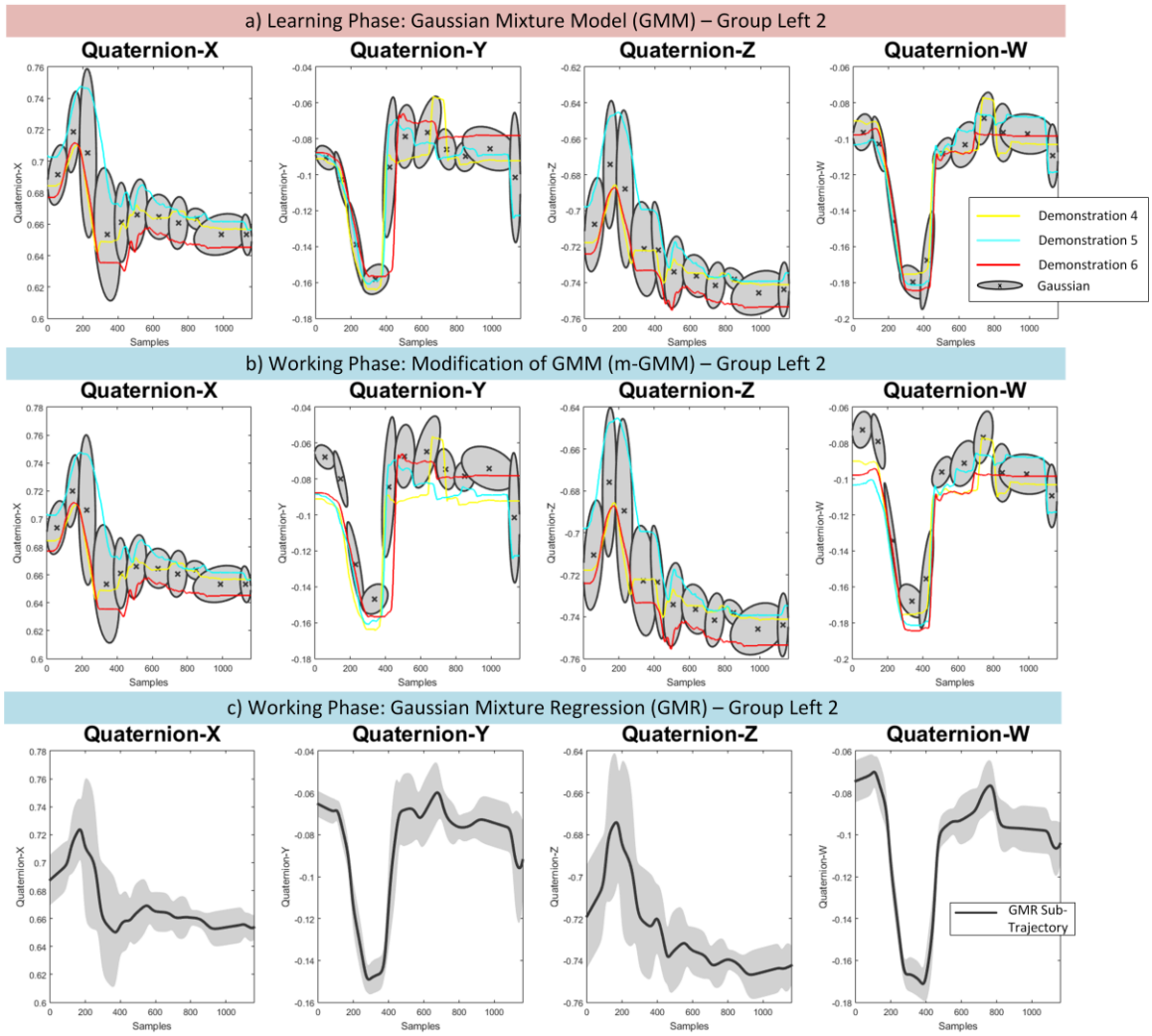


Figure 64: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 2 for the quaternions. b) The modification of the learned GMM for the sub-trajectories of group Left 2 during the trial 1 for the quaternions. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 2 for the quaternions.

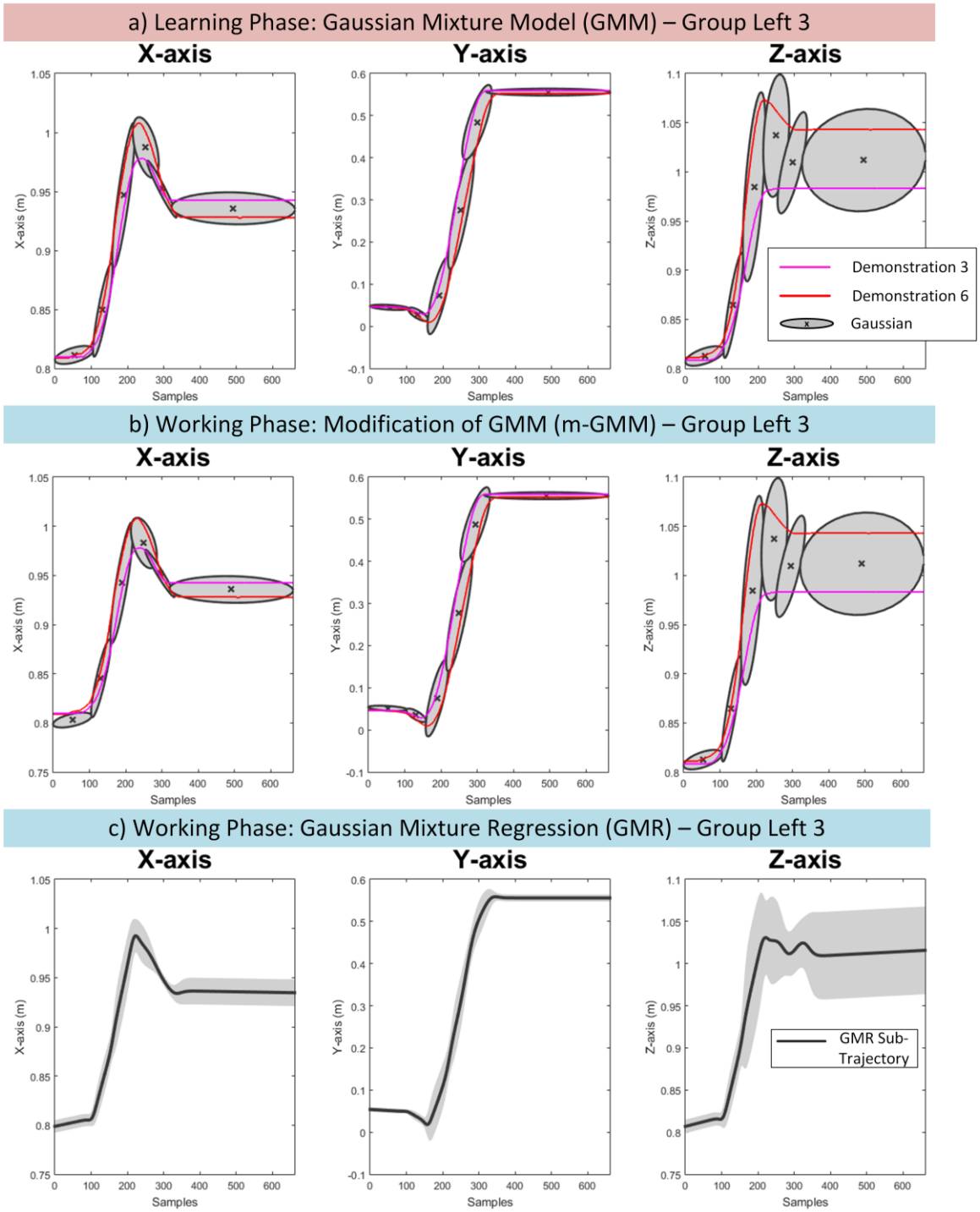


Figure 65: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 3 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 3 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 3 along the X-, Y-, and Z-axis.

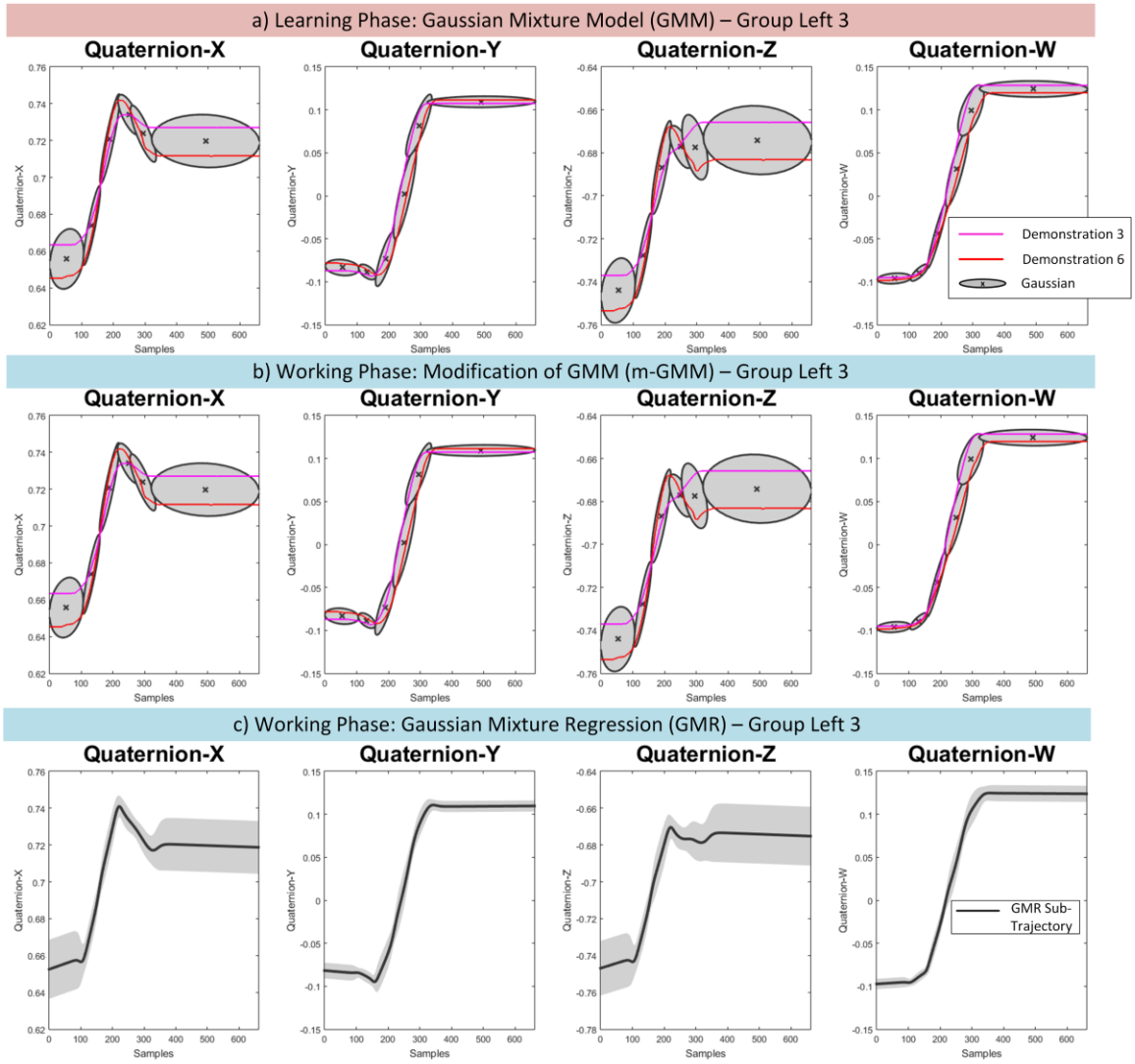


Figure 66: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 3 for the quaternions. b) The modification of the learned GMM for the sub-trajectories of group Left 3 during the trial 1 for the quaternions. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 3 for the quaternions.

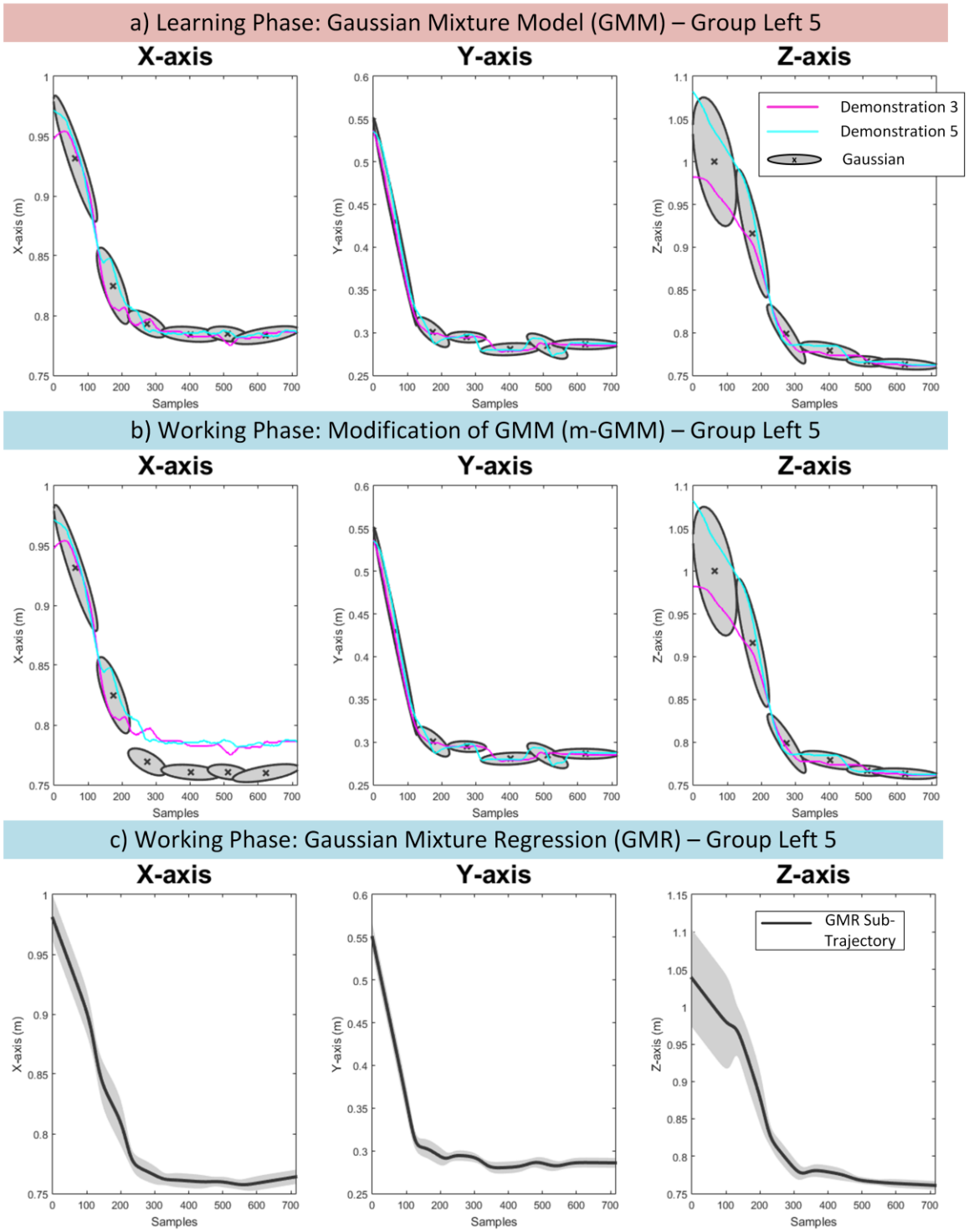


Figure 67: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 5 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 5 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 5 along the X-, Y-, and Z-axis.

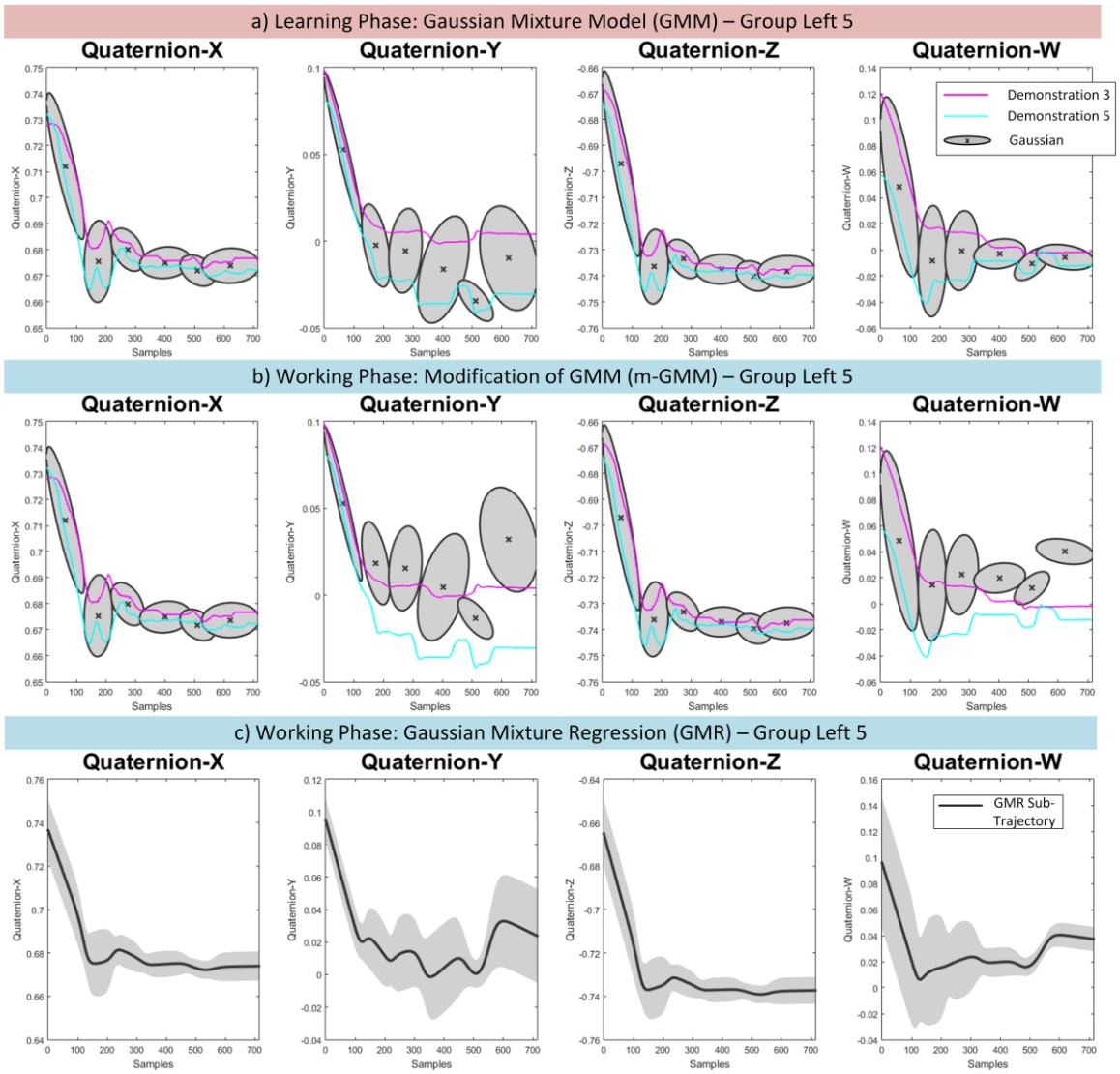


Figure 68: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 5 for the quaternions. b) The modification of the learned GMM for the sub-trajectories of group Left 5 during the trial 1 for the quaternions. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 5 for the quaternions.

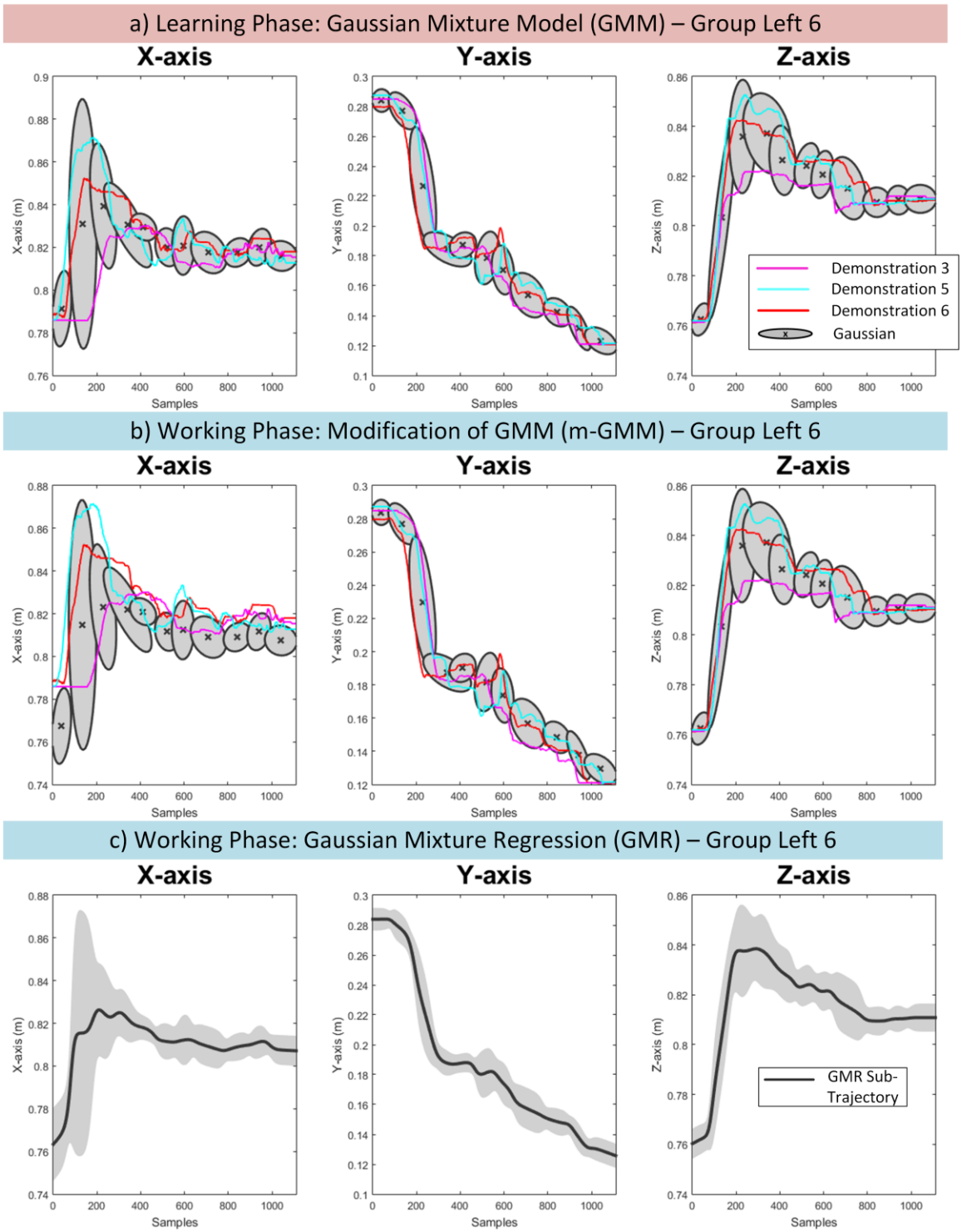


Figure 69: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 6 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 6 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 6 along the X-, Y-, and Z-axis.

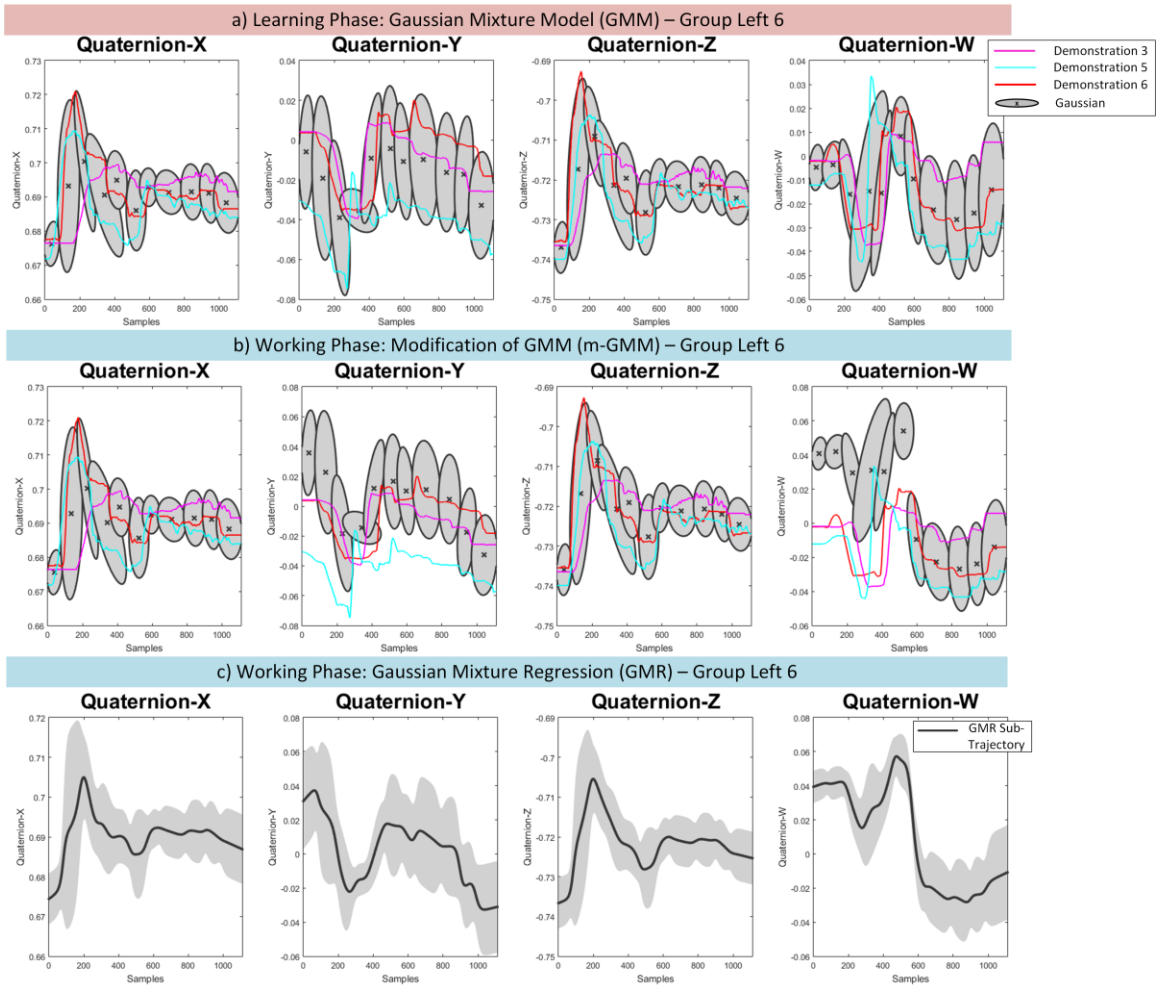


Figure 70: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 6 for the quaternions. b) The modification of the learned GMM for the sub-trajectories of group Left 6 during the trial 1 for the quaternions. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 6 for the quaternions.

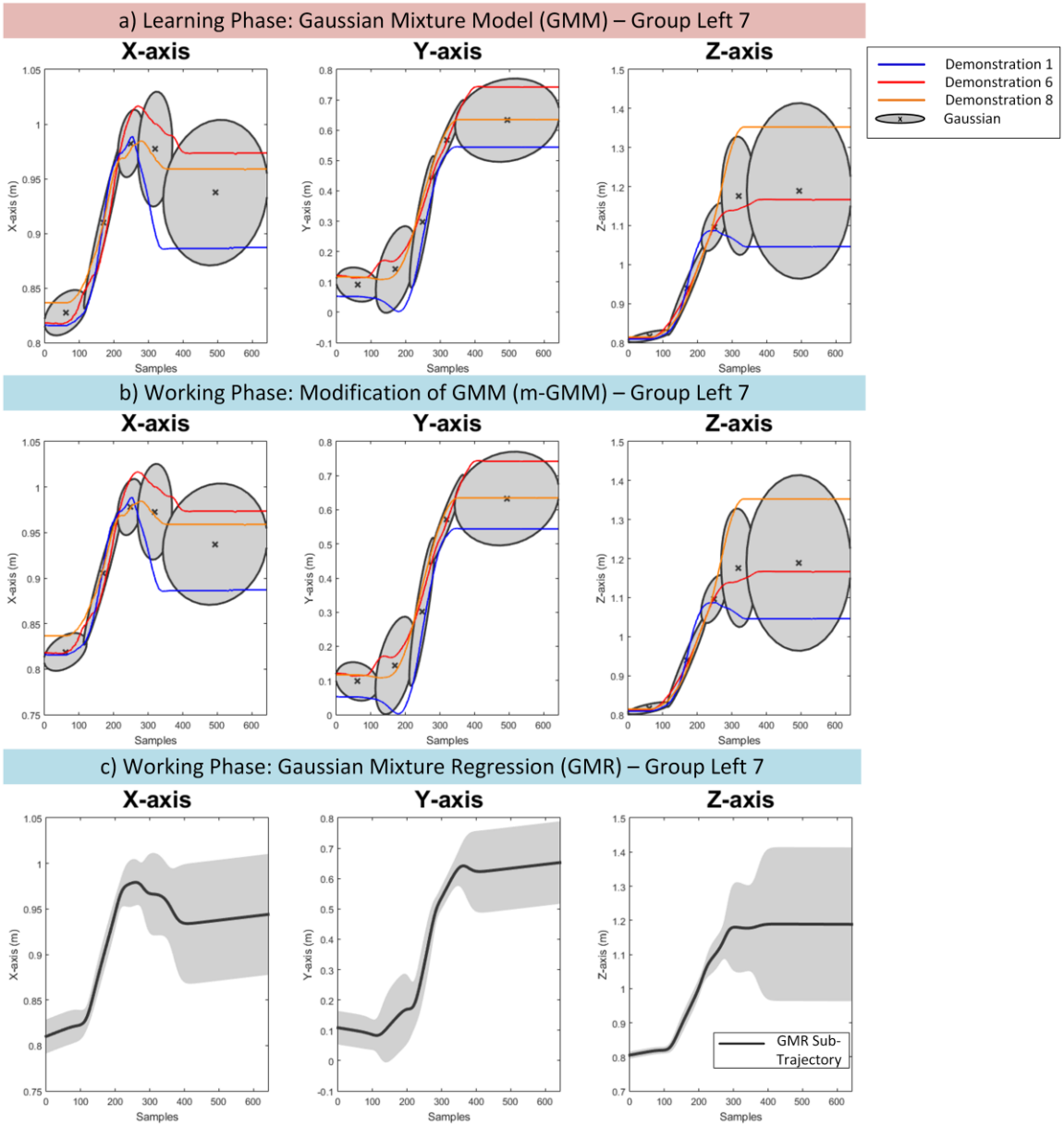


Figure 71: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 7 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 7 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 7 along the X-, Y-, and Z-axis.

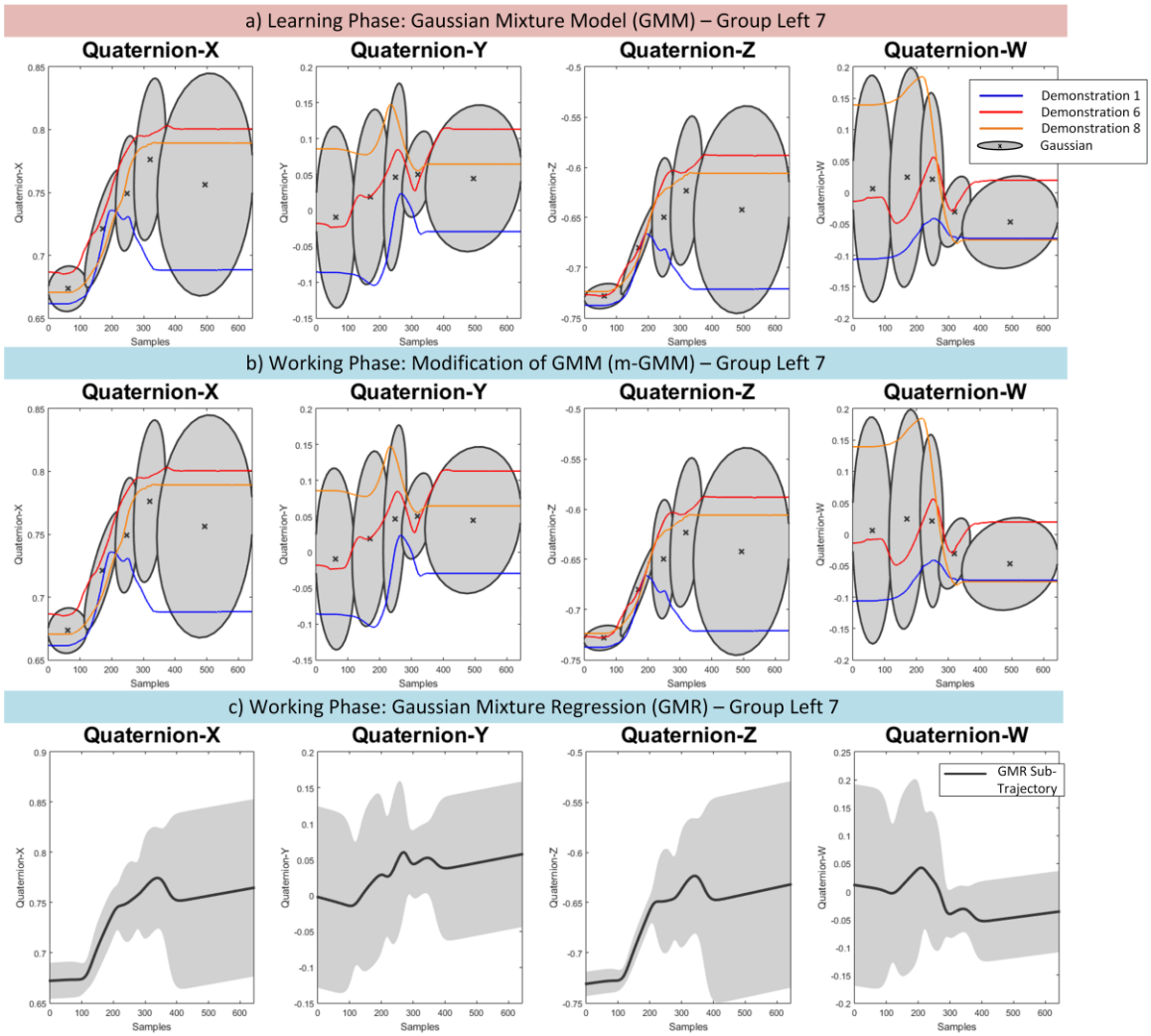


Figure 72: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Left 7 for the quaternions. b) The modification of the learned GMM for the sub-trajectories of group Left 7 during the trial 1 for the quaternions. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 7 for the quaternions.

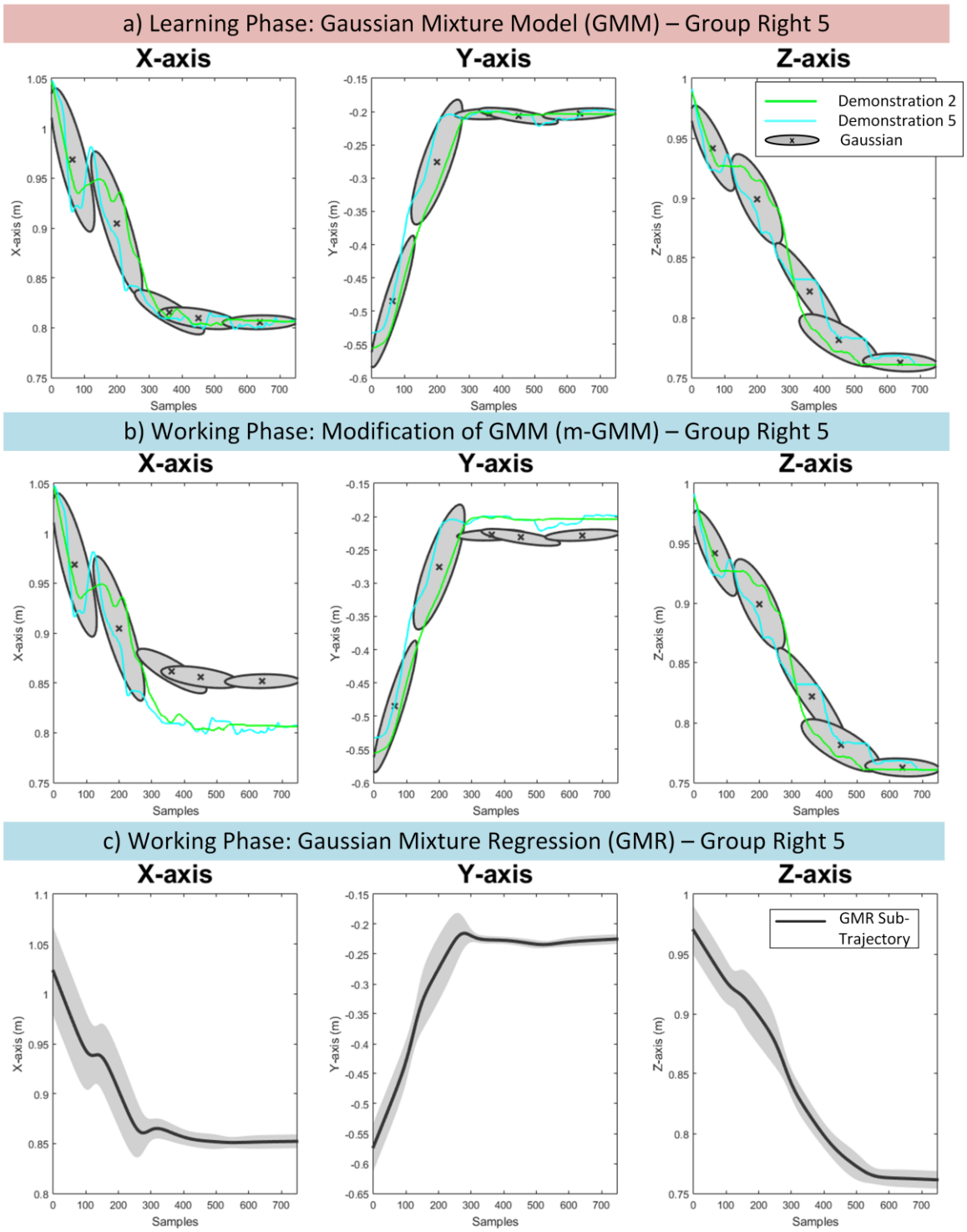


Figure 73: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Right 5 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Right 5 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 5 along the X-, Y-, and Z-axis.

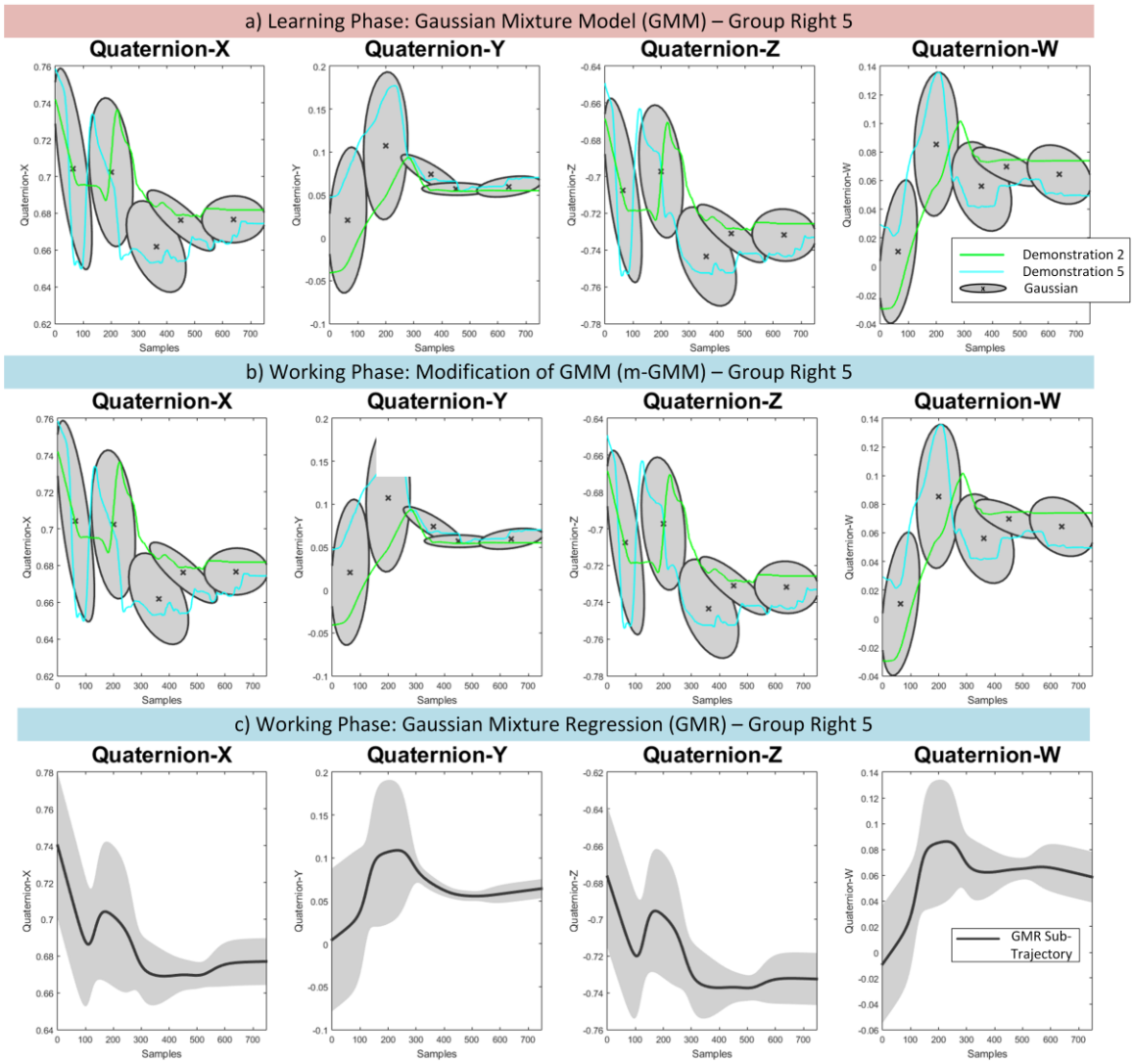


Figure 74: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Right 5 for the quaternions. b) The modification of the learned GMM for the sub-trajectories of group Right 5 during the trial 1 for the quaternions. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 5 for the quaternions.

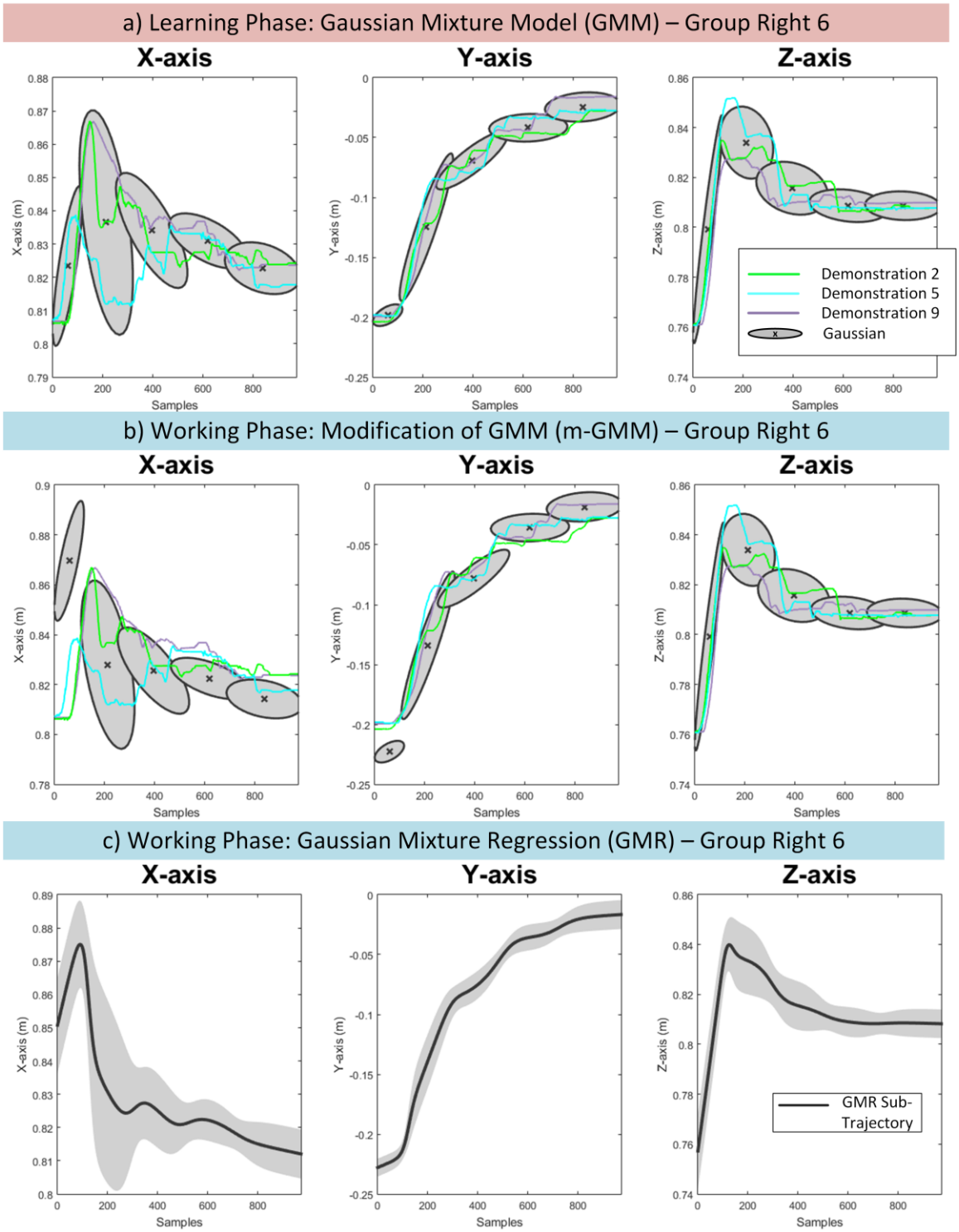


Figure 75: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Right 6 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Right 6 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 6 along the X-, Y-, and Z-axis.

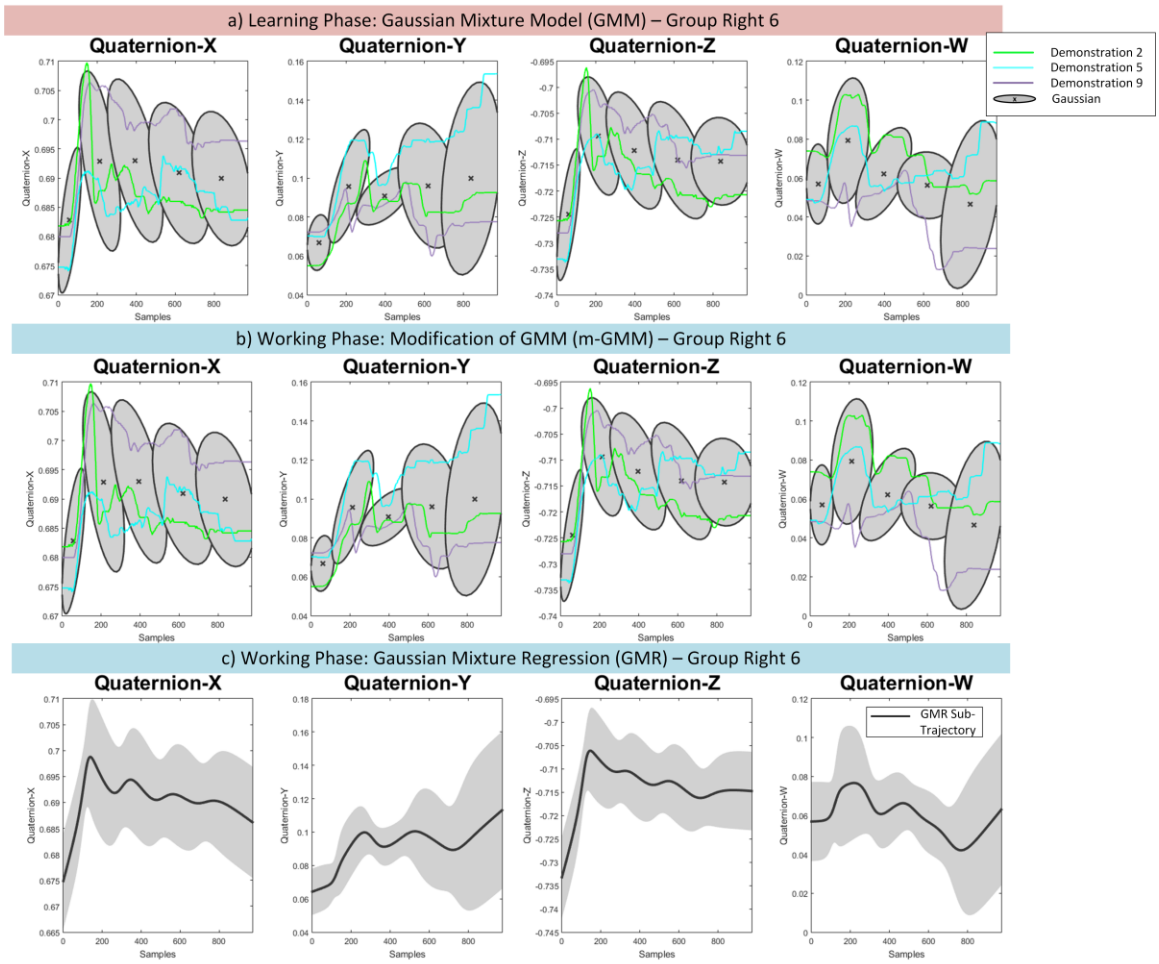


Figure 76: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Right 6 for the quaternions. b) The modification of the learned GMM for the sub-trajectories of group Right 6 during the trial 1 for the quaternions. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 6 for the quaternions.

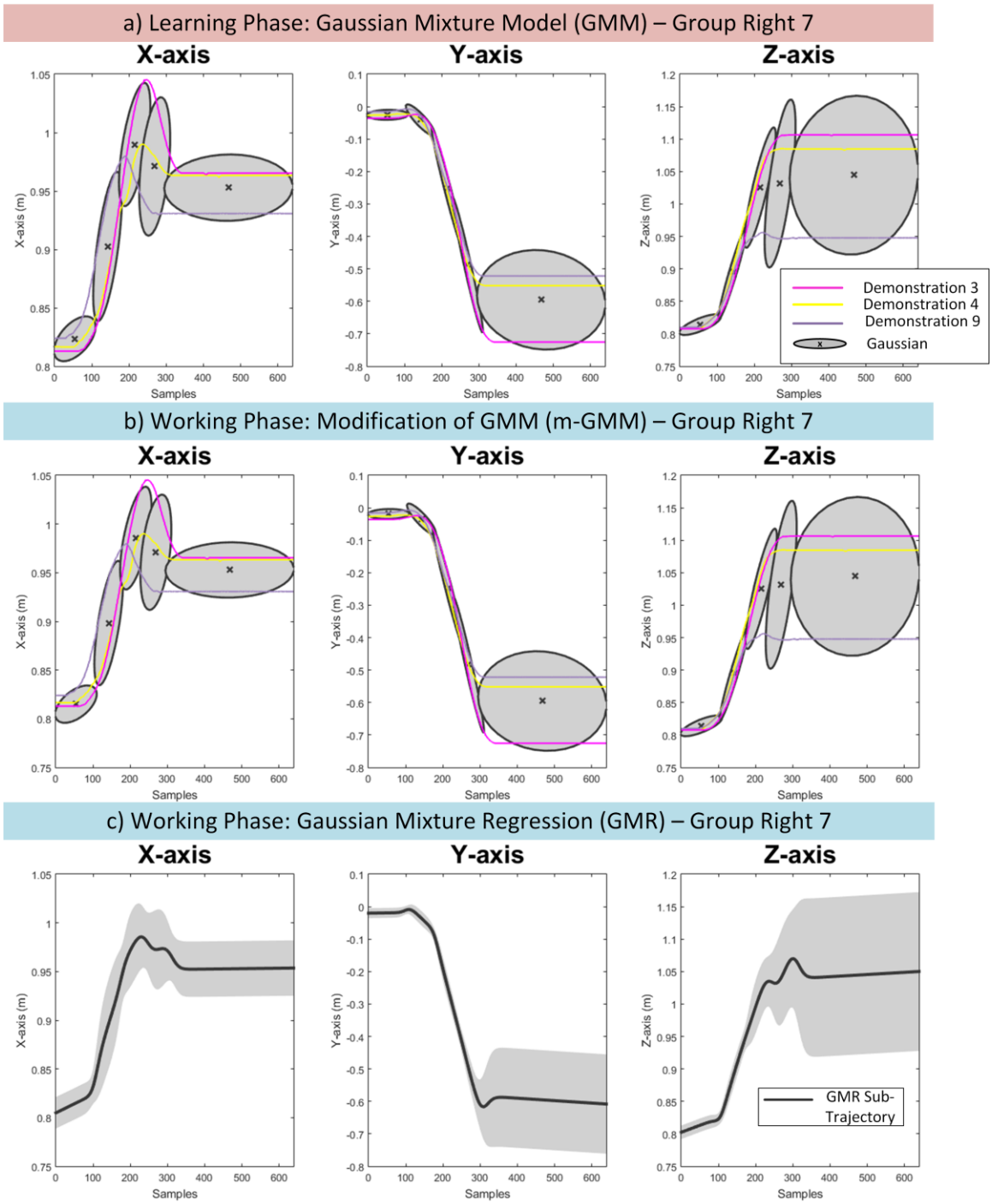


Figure 77: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Right 7 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Right 7 during the trial 1 along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 7 along the X-, Y-, and Z-axis.

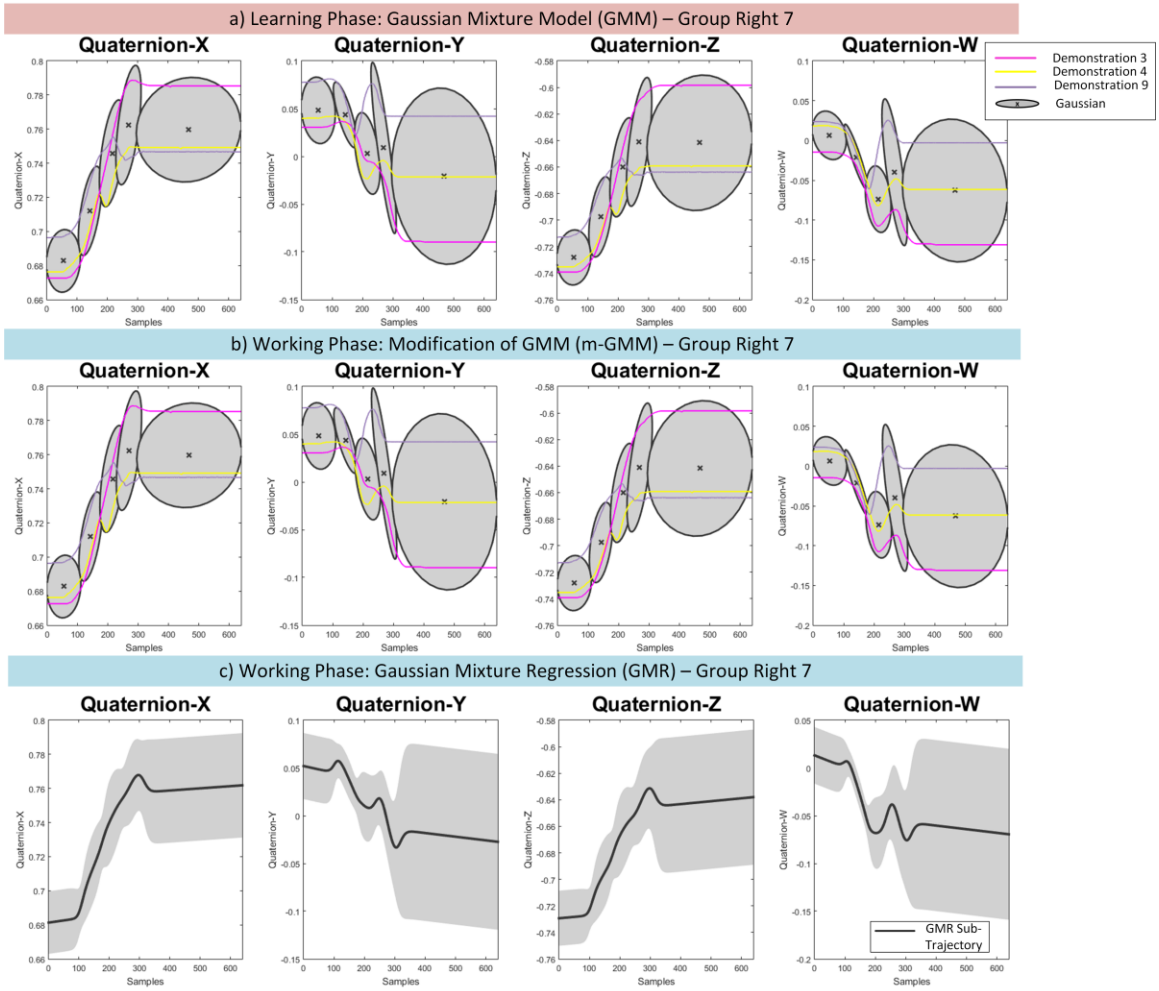


Figure 78: Robot gripper assembly task. a) The learned GMM for the sub-trajectories of group Right 7 for the quaternions. b) The modification of the learned GMM for the sub-trajectories of group Right 7 during the trial 1 for the quaternions. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 7 for the quaternions.

II. 'Pins into Holes' Task (Task 2)

In this section, the complete results for the task 'Pins into Holes' (section 4.3) are presented. Table 24 associates the groups for the robot gripper assembly tasks with the figure number, where the learned GMM, m-GMM and GMR are illustrated. In the figures, the *x*-, *y*-, *z*- dimensions are denoted as X-,Y-,Z-axis, respectively.

Table 24: Association between the groups of sub-trajectories, the high-level actions and the figures for the task 'Pins into Holes' (Task 2).

Group	**Sub-trajectories between the high-level actions**	**Figure GMM/m-GMM/GMR for *x*-, *y*-, *z*- dimensions**
Right 1	*RV Home – Home* and *RV Close – Color box info*	Figure 79
Right 2	*RV Close – Color box info* and *RV Open – Red holder info*	Figure 80
Right 3	*RV Open – Red holder info* and *RV Home – Home*	Figure 81
Left 5	*L2 Home – Home* and *L2 Close – Pin 10 info*	Figure 82
Left 6	*L2 Close – Pin 10 info* and *L2 Open – Color box info*	Figure 83
Left 7	*L2 Open – Color box info* and *L2 Home – Home*	Figure 84
Left 8 – version 1	*L2 Home – Home* and *L2 Close – Pin 8 info*	Figure 85
Left 9 – version 1	*L2 Close – Pin 8 info* and *L2 Open – Color box info*	Figure 86
Left 10 – version 1	*L2 Open – Color box info* and *L2 Home – Home*	Figure 87
Left 8 – version 2	*L2 Home – Home* and *L2 Close – Pin 6 info*	Figure 40
Left 9 – version 2	*L2 Close – Pin 6 info* and *L2 Open – Color box info*	Figure 41
Left 10 – version 2	*L2 Open – Color box info* and *L2 Home – Home*	Figure 42

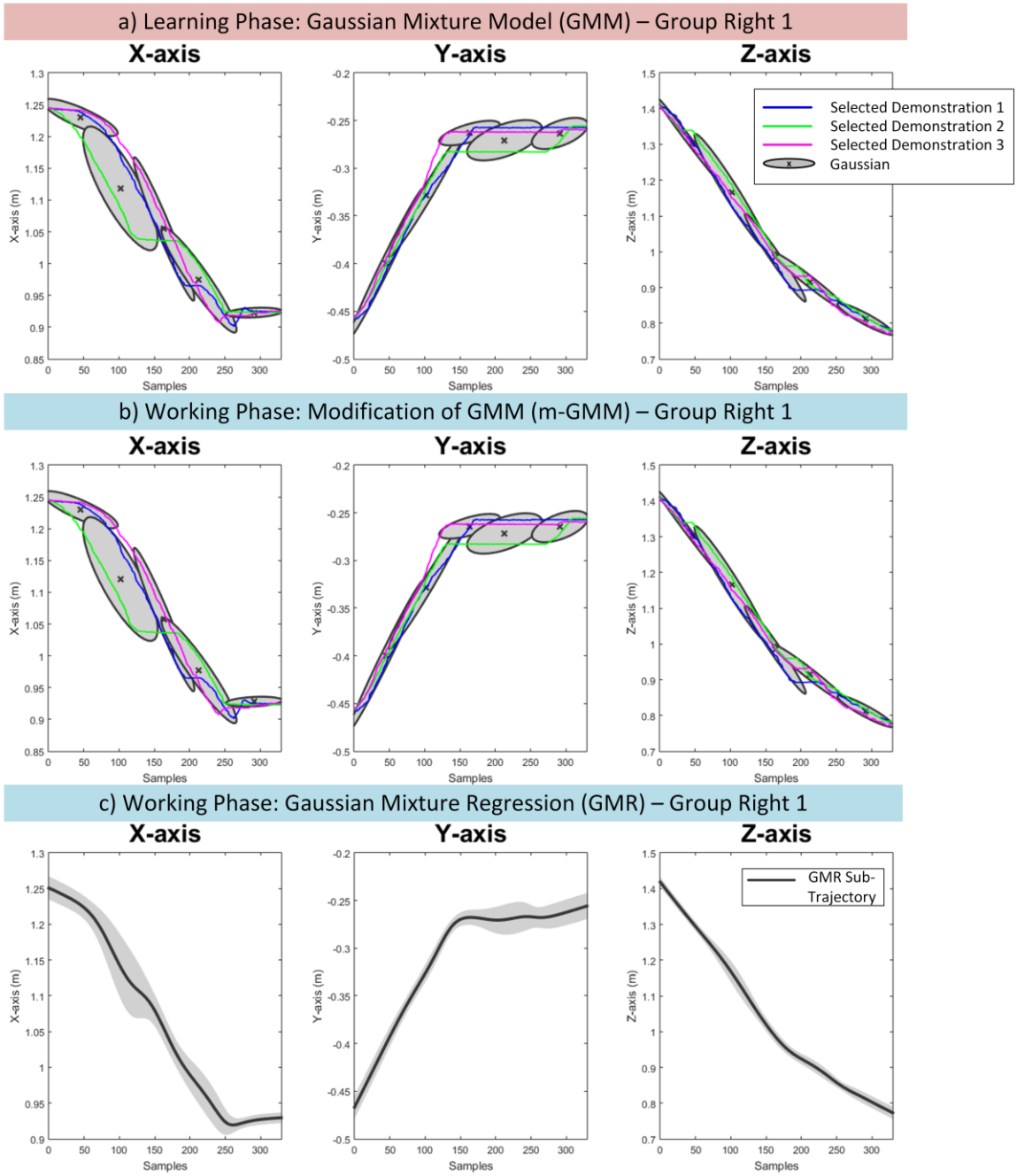


Figure 79: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Right 1 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Right 1 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 1 along the X-, Y-, and Z-axis.

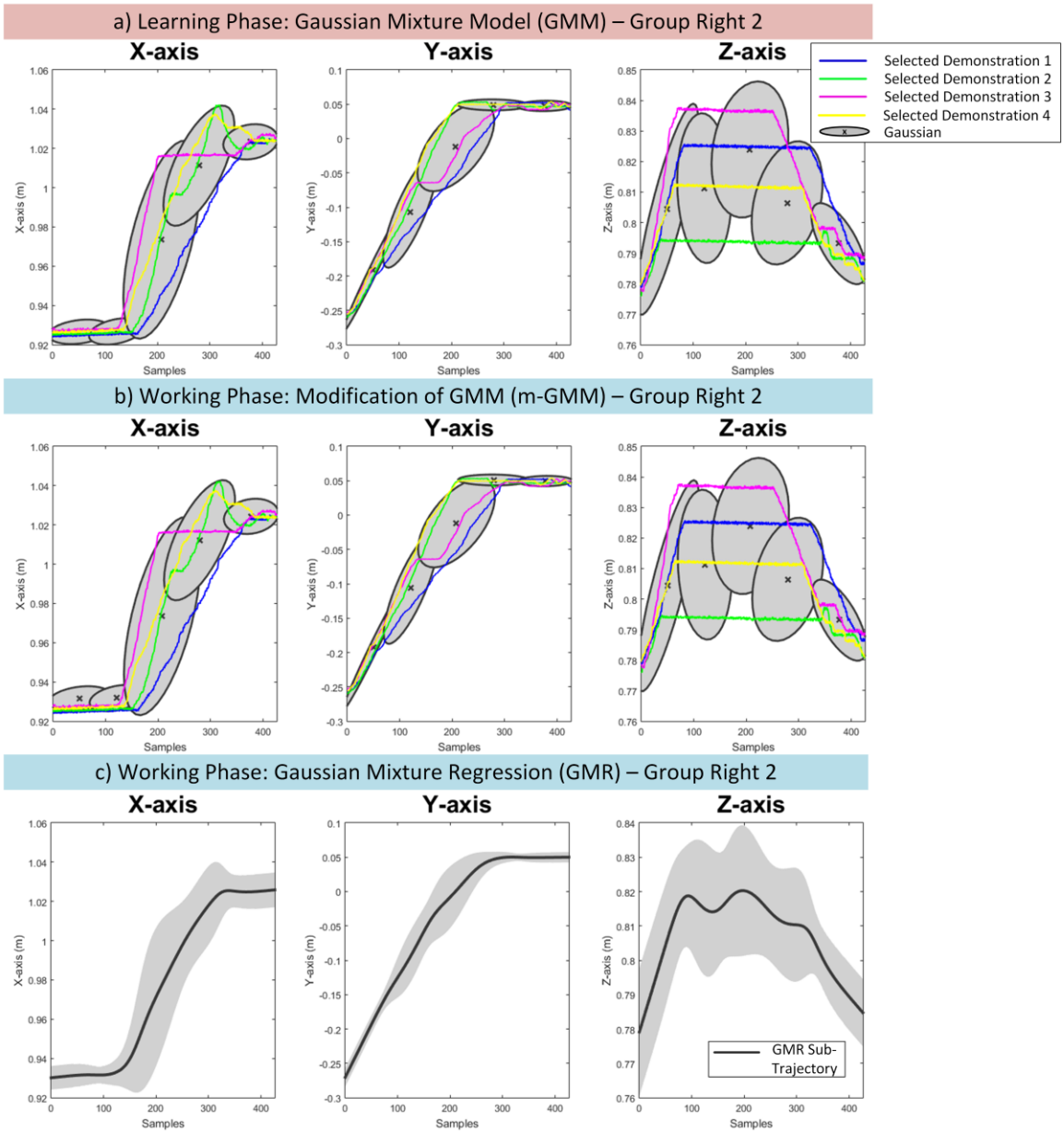


Figure 80: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Right 2 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Right 2 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 2 along the X-, Y-, and Z-axis.

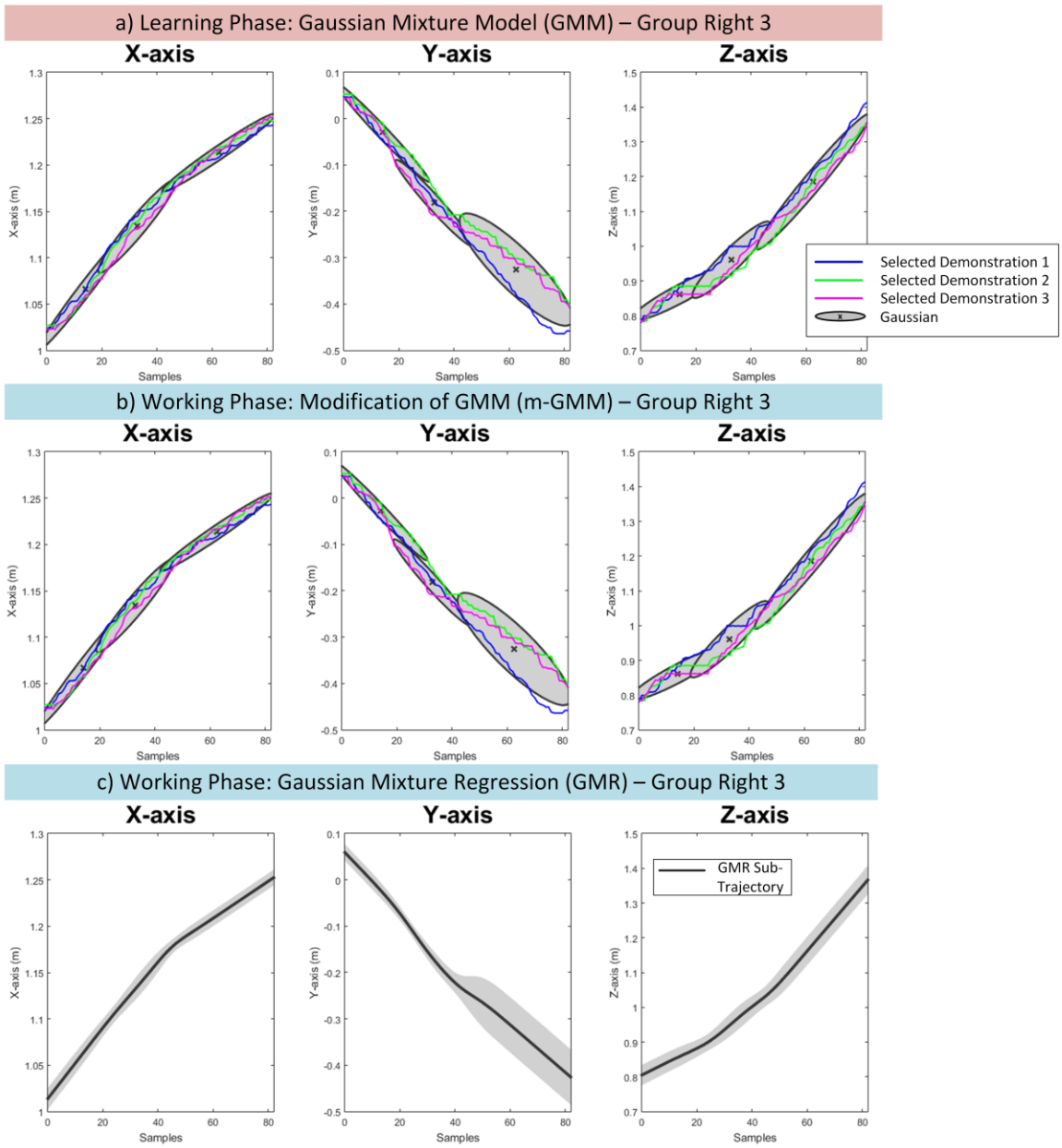


Figure 81: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Right 3 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Right 3 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Right 3 along the X-, Y-, and Z-axis.

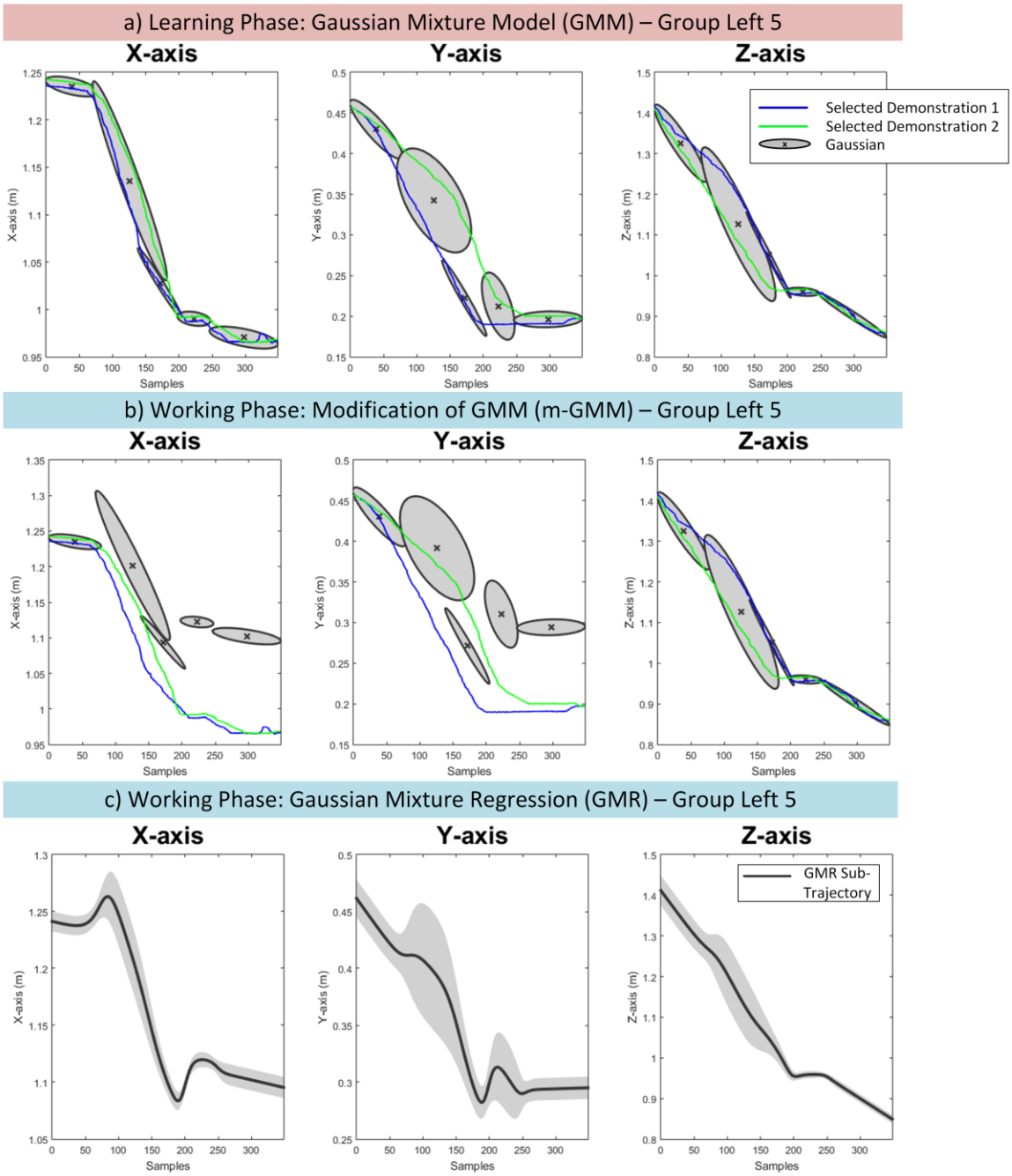


Figure 82: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Left 5 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 5 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 5 along the X-, Y-, and Z-axis.

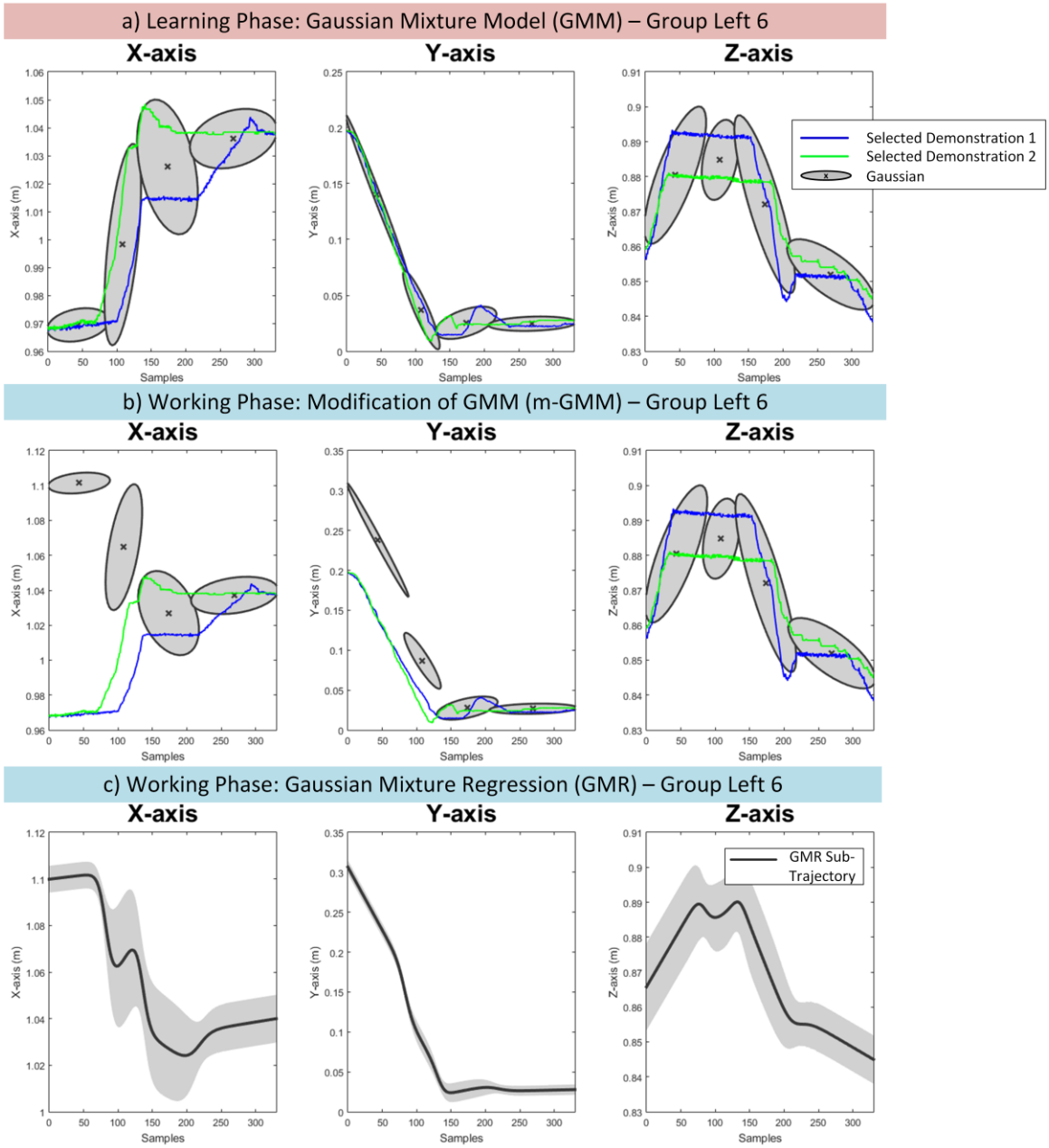


Figure 83: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Left 6 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 6 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 6 along the X-, Y-, and Z-axis.

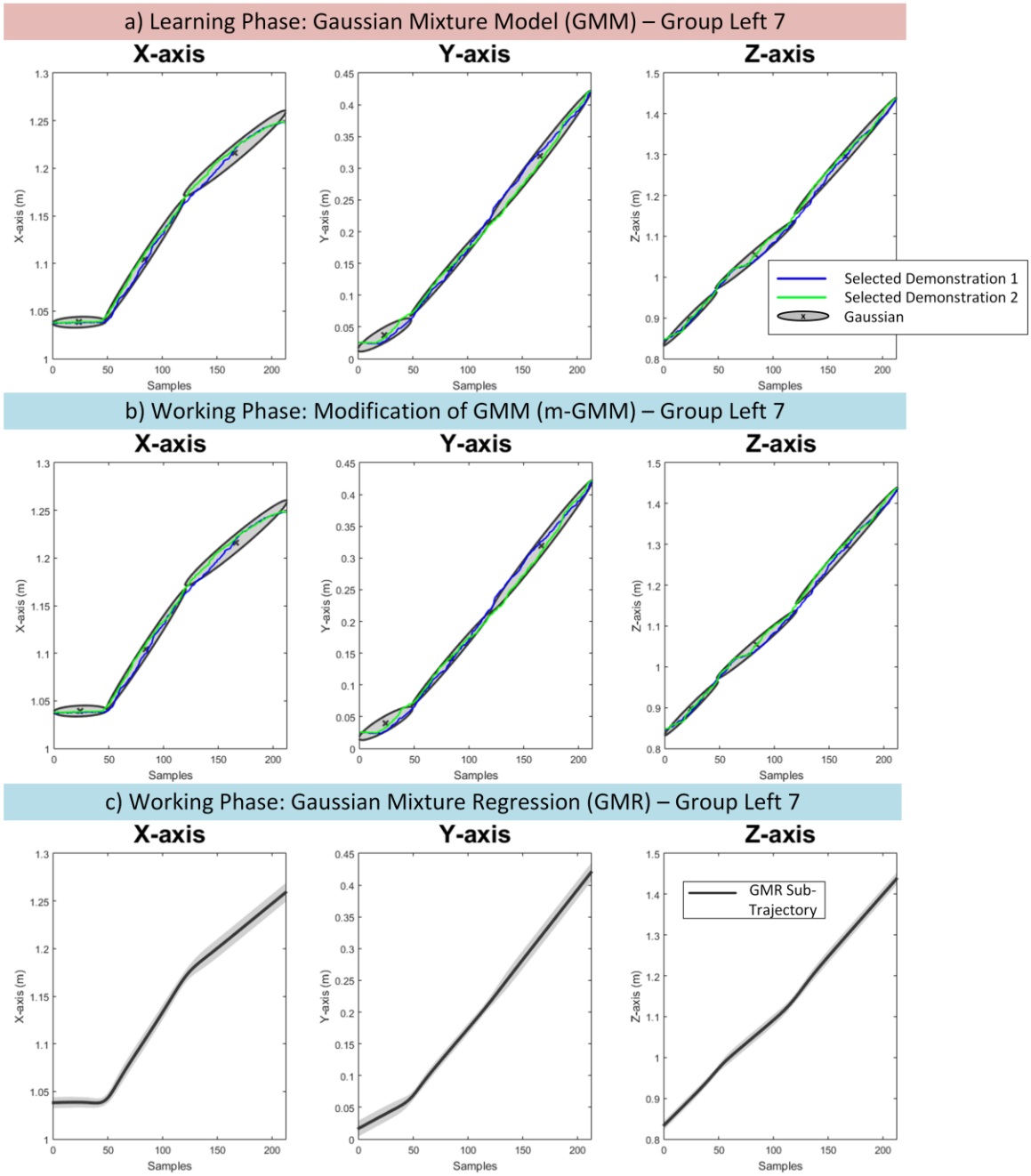


Figure 84: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Left 7 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 7 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 7 along the X-, Y-, and Z-axis.

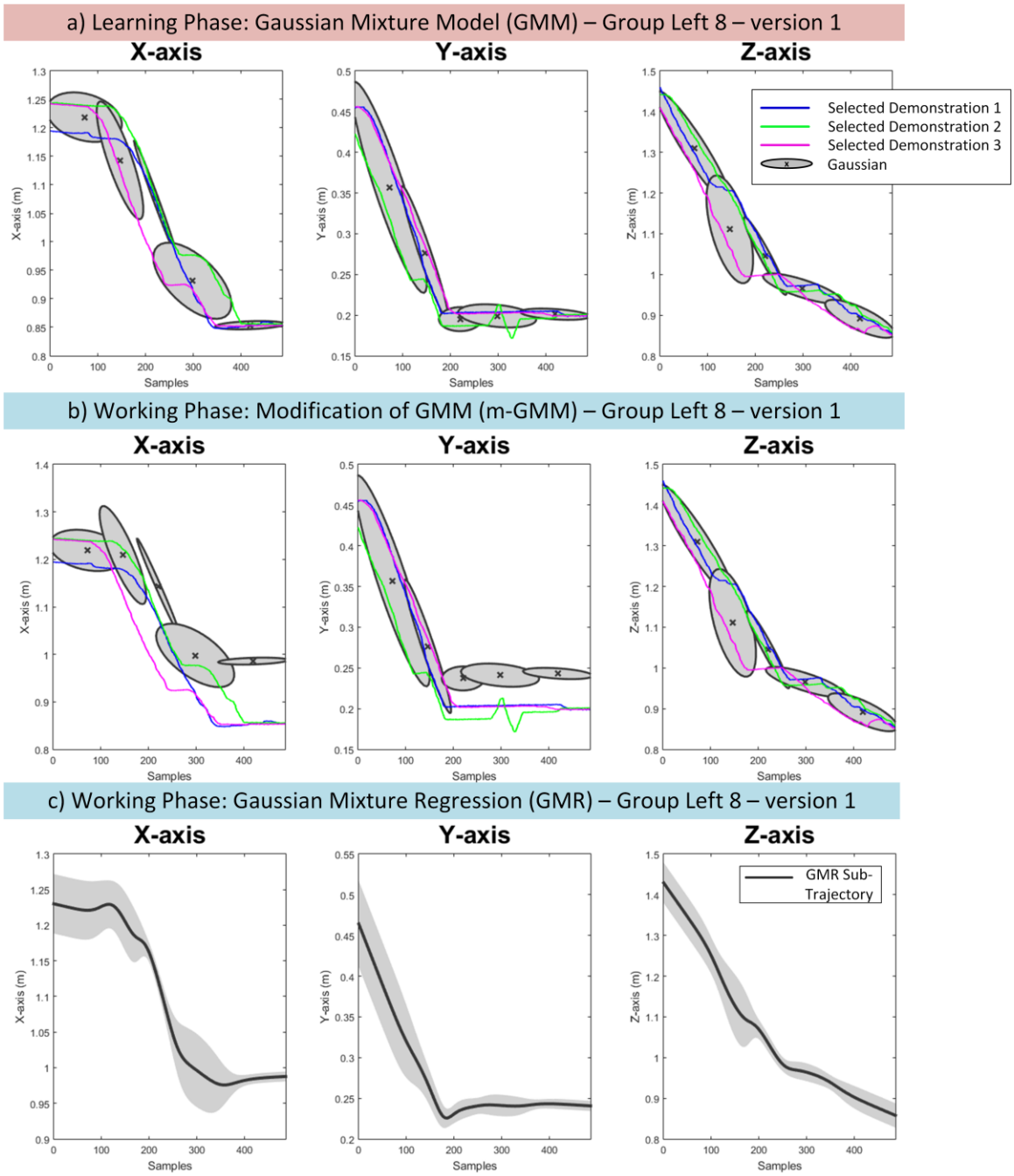


Figure 85: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Left 8 – version 1 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 8 – version 1 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 8 – version 1 along the X-, Y-, and Z-axis.

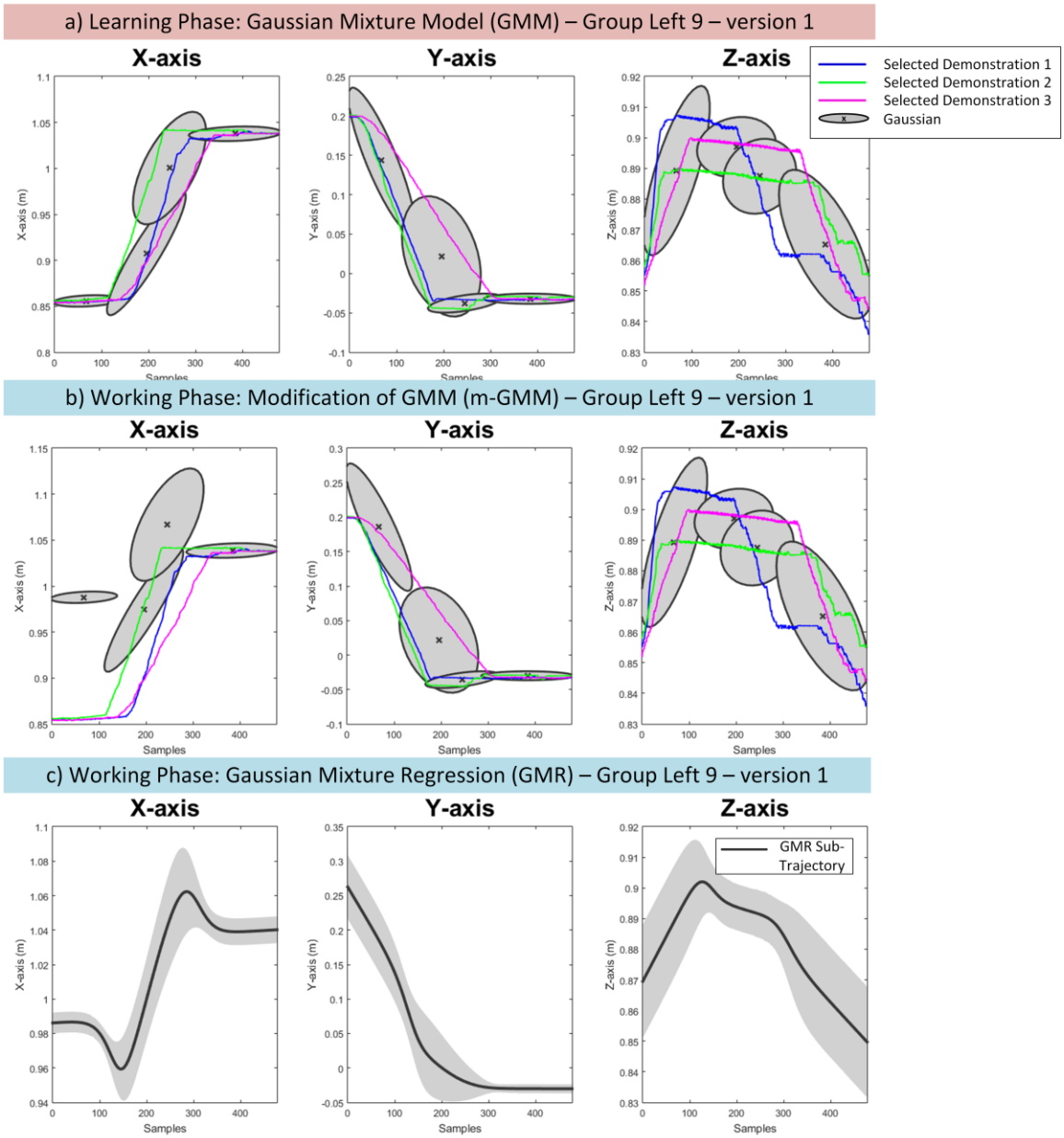


Figure 86: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Left 9 – version 1 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 9 – version 1 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 9 – version 1 along the X-, Y-, and Z-axis.

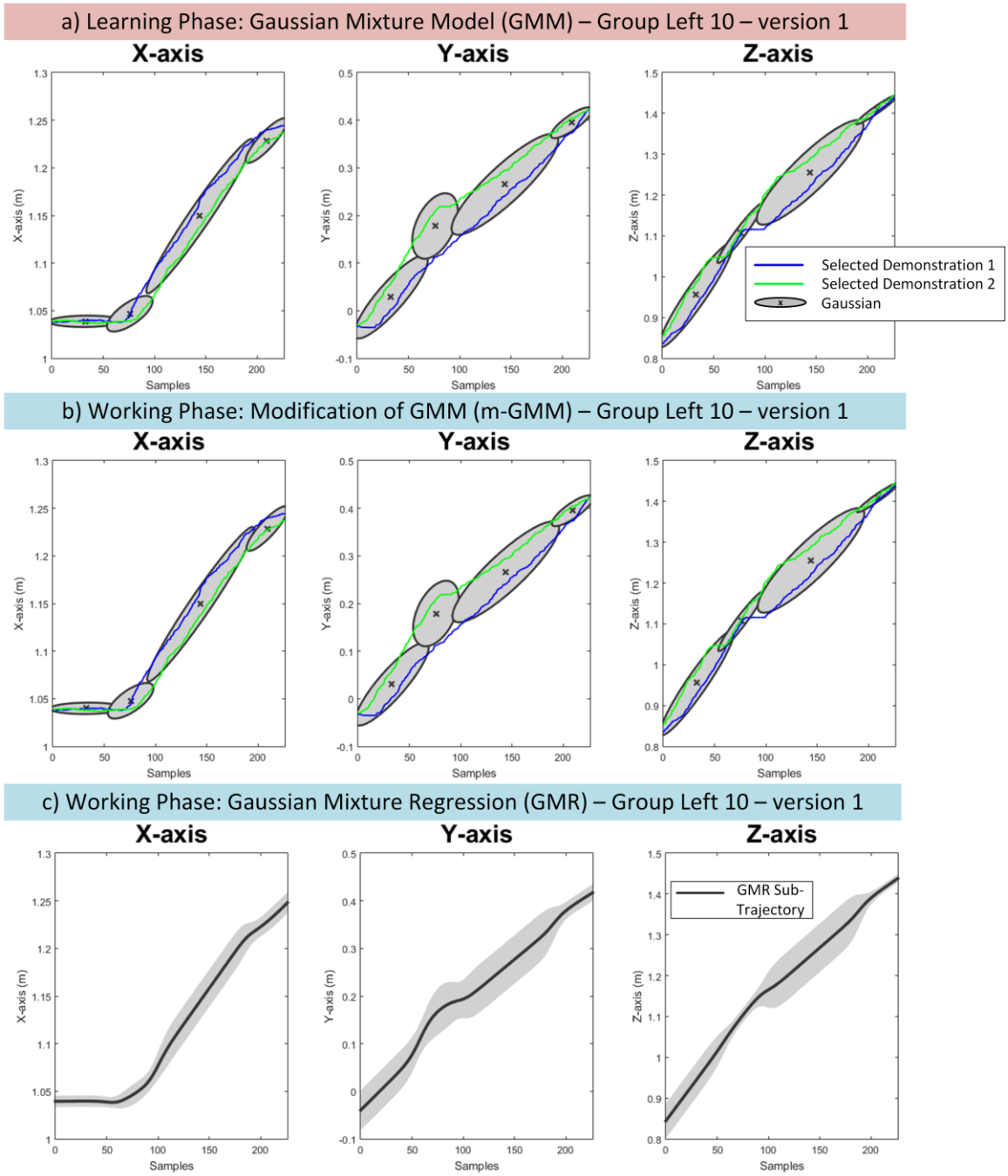


Figure 87: Task 'Pins into Holes'. a) The learned GMM for the sub-trajectories of group Left 10 – version 1 along the X-, Y-, and Z-axis. b) The modification of the learned GMM for the sub-trajectories of group Left 10 – version 1 during the working phase along the X-, Y-, and Z-axis. c) The generated GMR sub-trajectory produced by the m-GMM of group Left 10 – version 1 along the X-, Y-, and Z-axis.

Appendix D: Experimental Results in Assistive Robotics Application

In this section, the complete results for the application in assistive robotics (Chapter 5) are presented. Figure 88 and Figure 89 respectively show the first and second sub-trajectory of the demonstration provided by the tetraplegic user and the learned GMM for all 7 dimensions. The *x*-,*y*-,*z*- dimensions are denoted as X-,Y-,Z-axis, respectively. The quaternions *qx, qy, qz, qw* are denoted as Quaternion-X,-Y,-Z,-W, respectively.

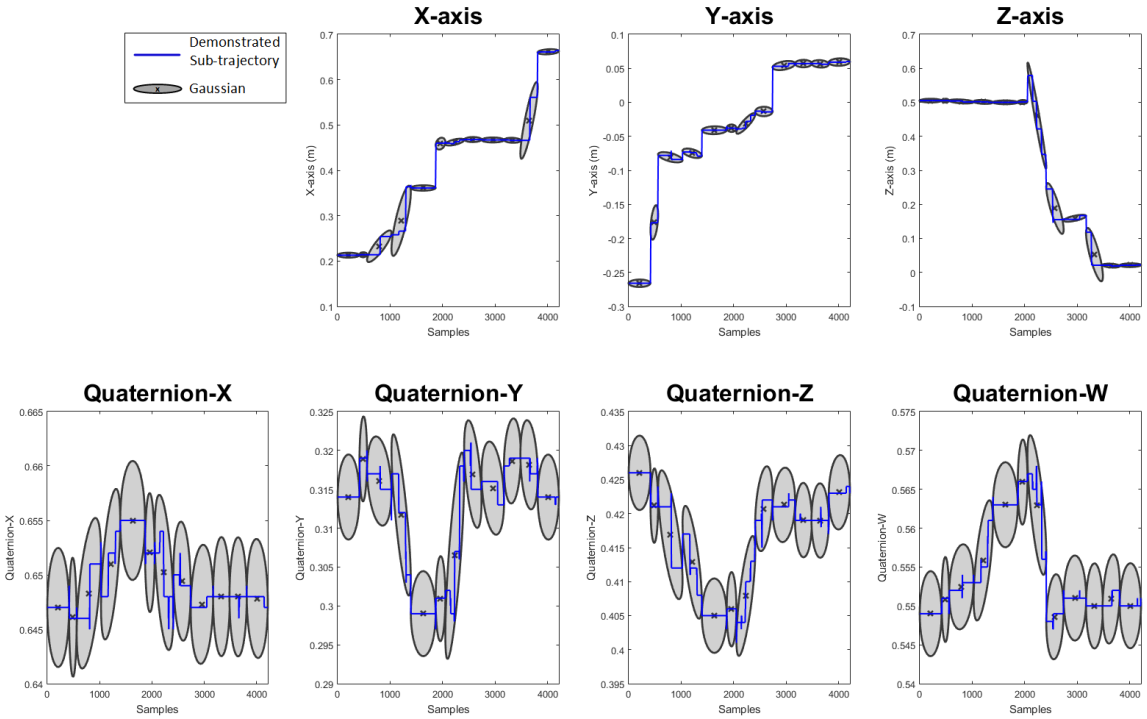


Figure 88: Assistive Task - The learned GMM for the first sub-trajectory in 7 dimensions.

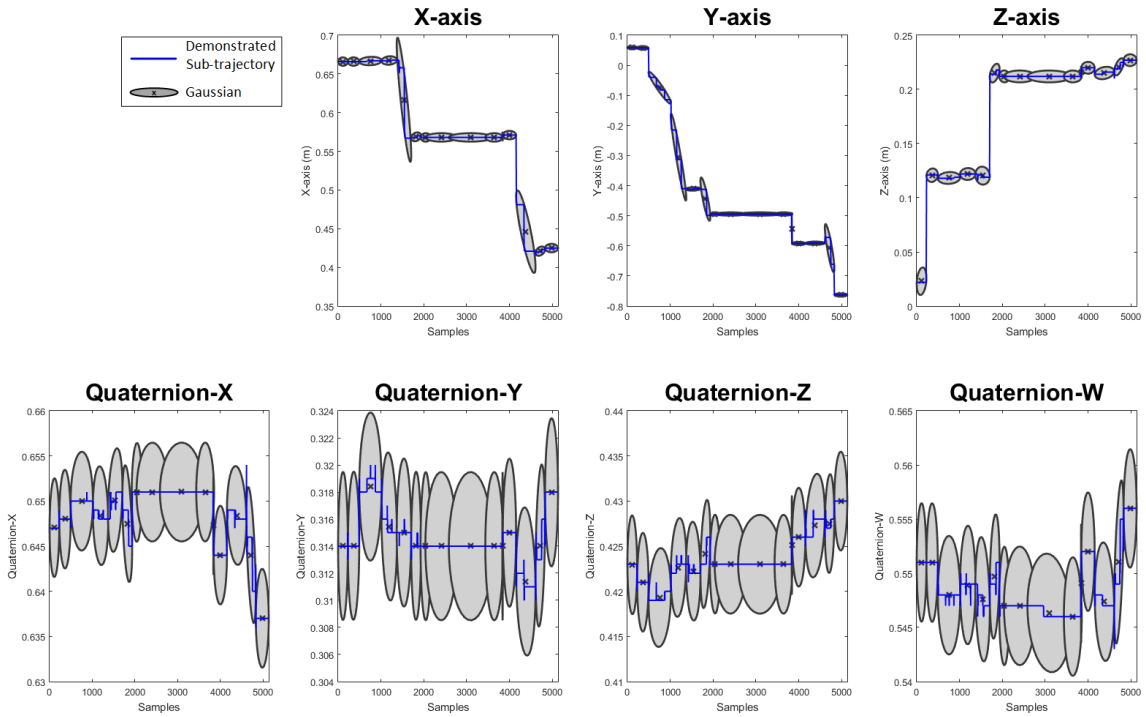


Figure 89: Assistive Task - The learned GMM for the second sub-trajectory in 7 dimensions.

Three trials are performed in robot working phase to evaluate the presented framework. In each trial, the position of the person and the cup differ from the demonstration, as shown in Table 20. The m-GMM/GMR outputs of all 7 dimensions for each trial are presented as follows.

- **Trial 1:** The output of m-GMM/GMR for the first sub-trajectory

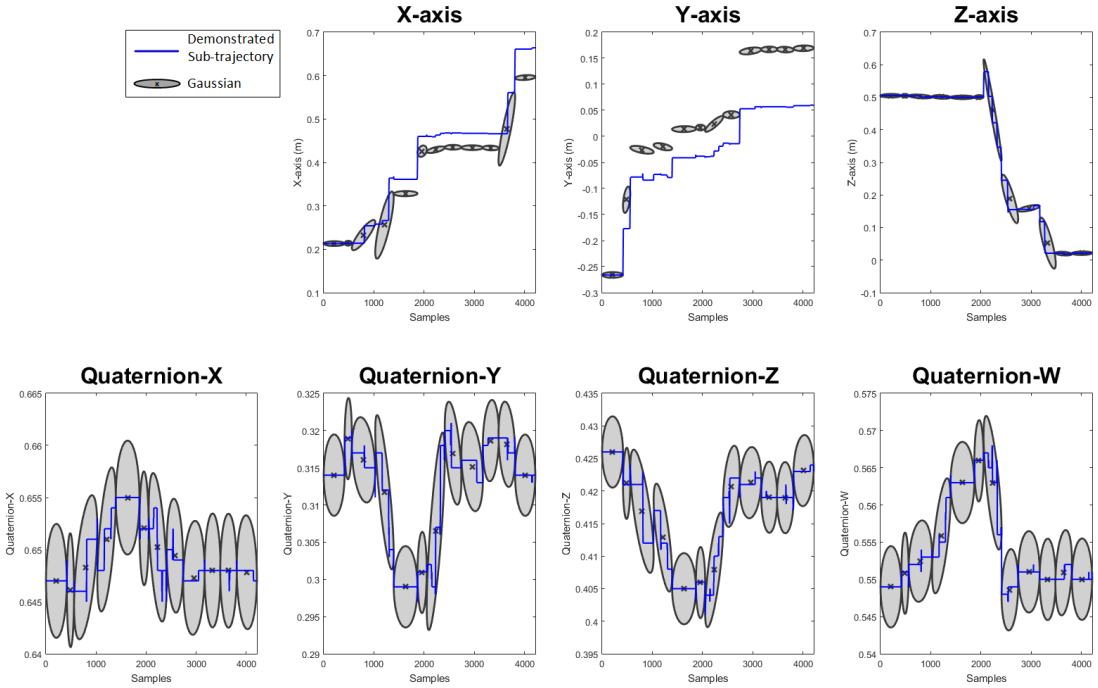


Figure 90: Assistive Task - The modification of the GMM for the first sub-trajectory – Trial 1

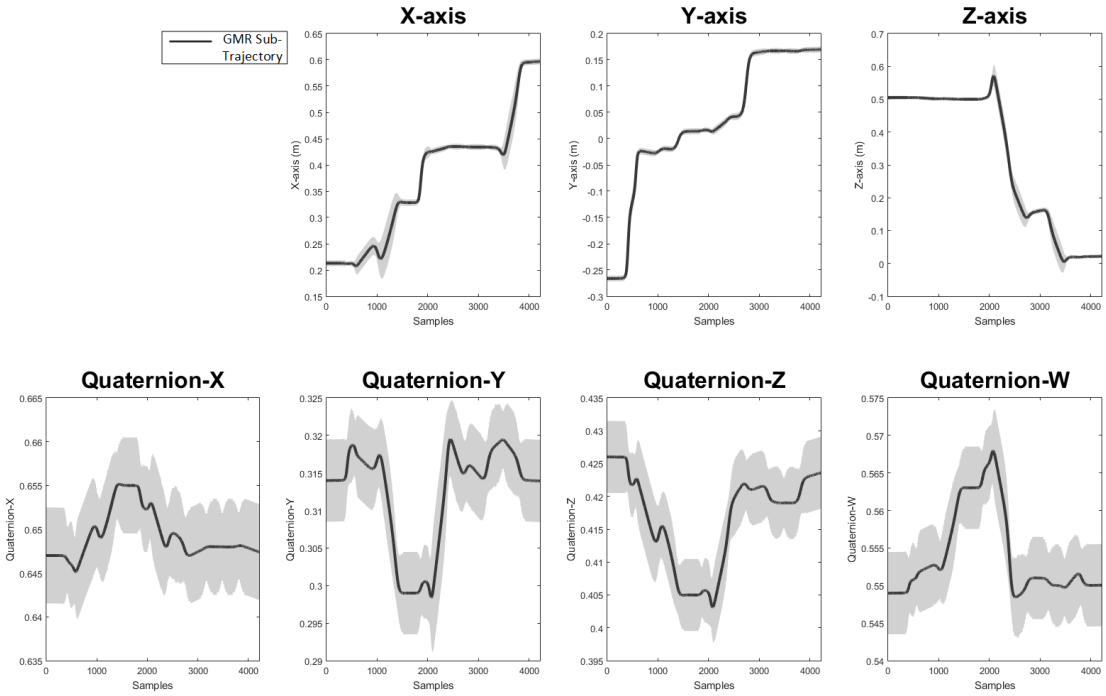


Figure 91: Assistive Task - The first GMR sub-trajectory produced by the m-GMM – Trial 1

- **Trial 1:** The output of m-GMM/GMR for the second sub-trajectory

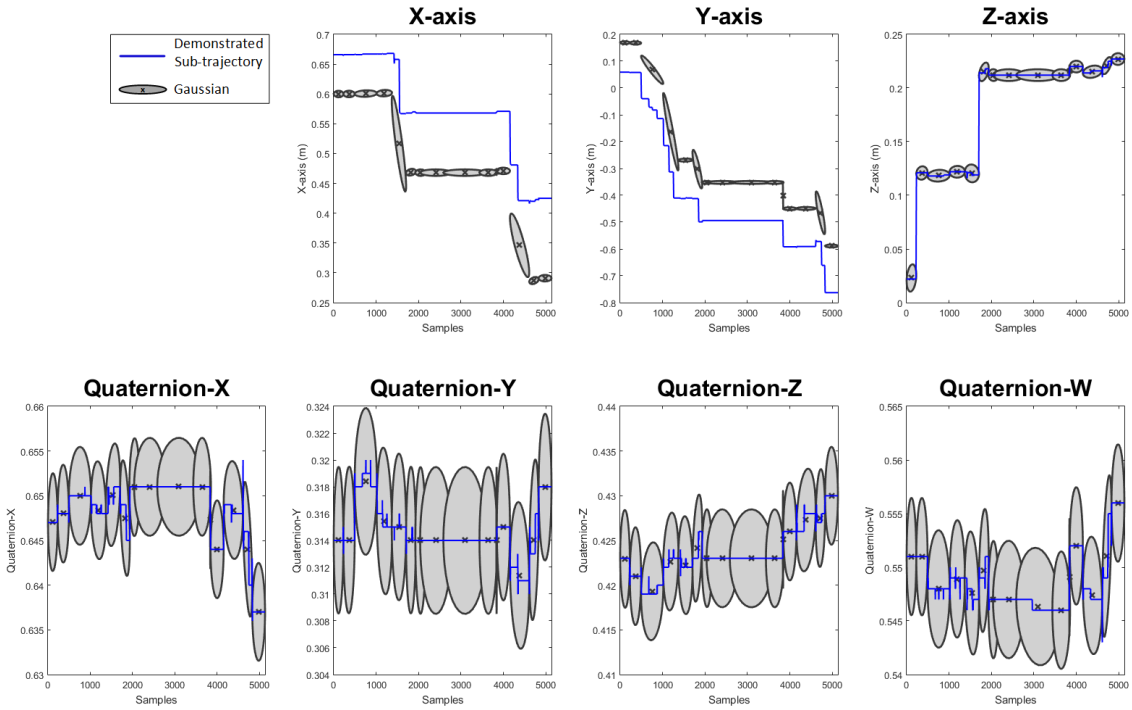


Figure 92: Assistive Task - The modification of the GMM for the second sub-trajectory – Trial 1

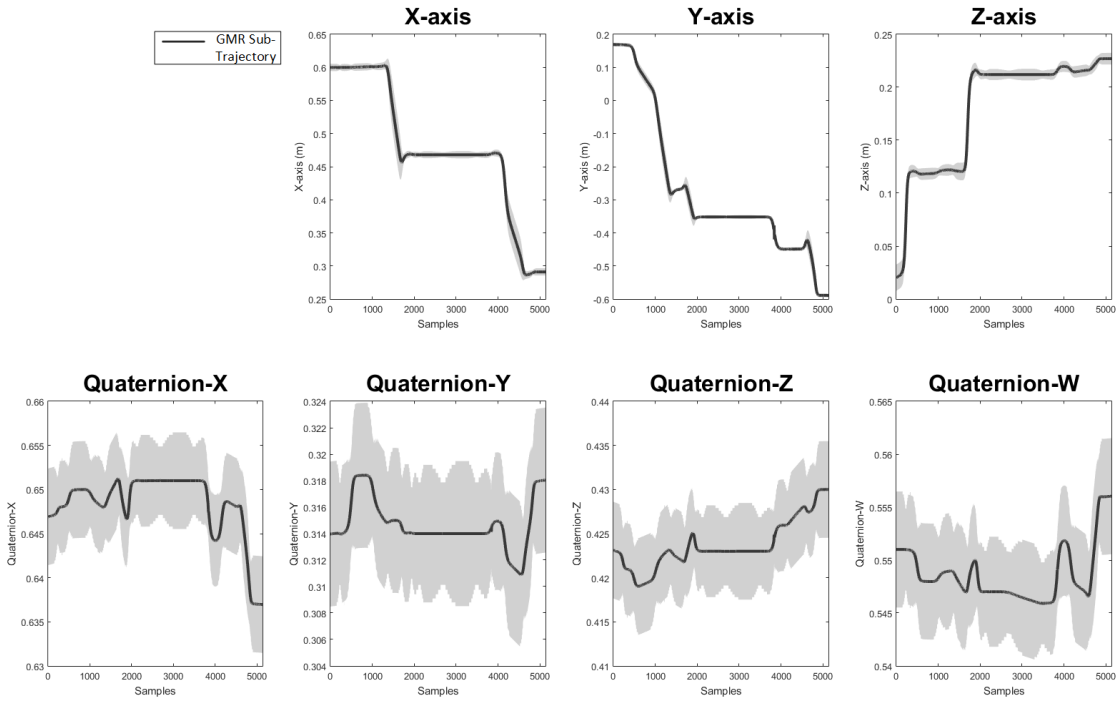


Figure 93: Assistive Task - The second GMR sub-trajectory produced by the m-GMM – Trial 1

- **Trial 2:** The output of m-GMM/GMR for the first sub-trajectory

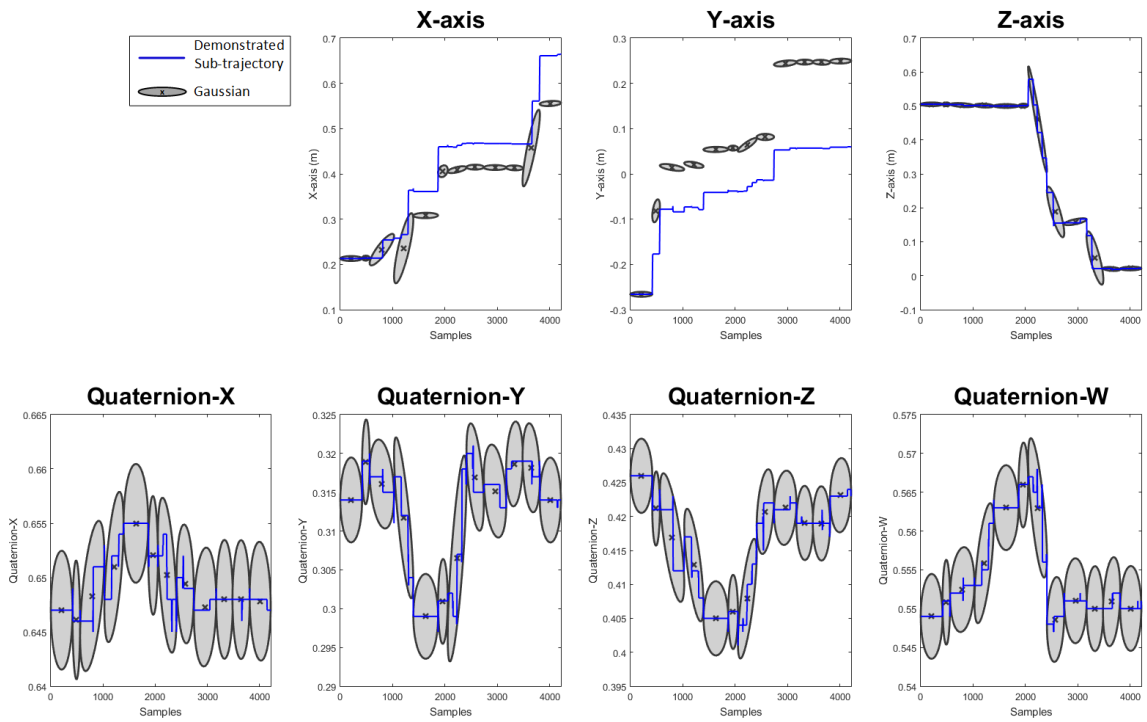


Figure 94: Assistive Task - The modification of the GMM for the first sub-trajectory – Trial 2

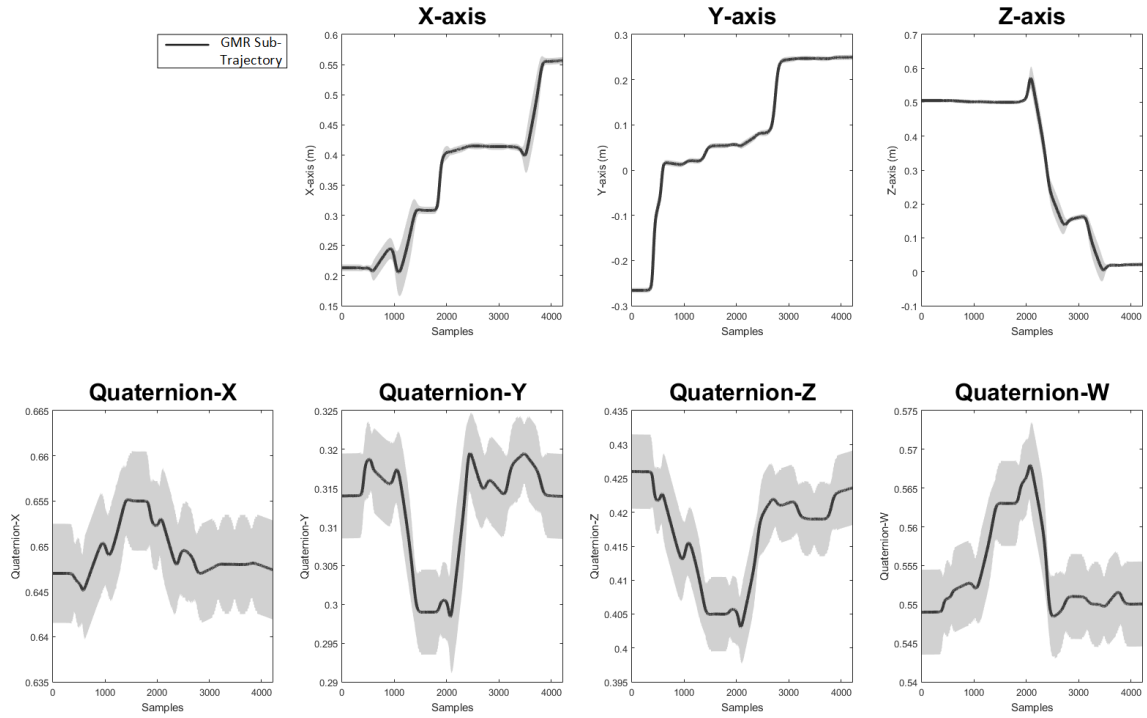


Figure 95: Assistive Task - The first GMR sub-trajectory produced by the m-GMM – Trial 2

- **Trial 2:** The output of m-GMM/GMR for the second sub-trajectory

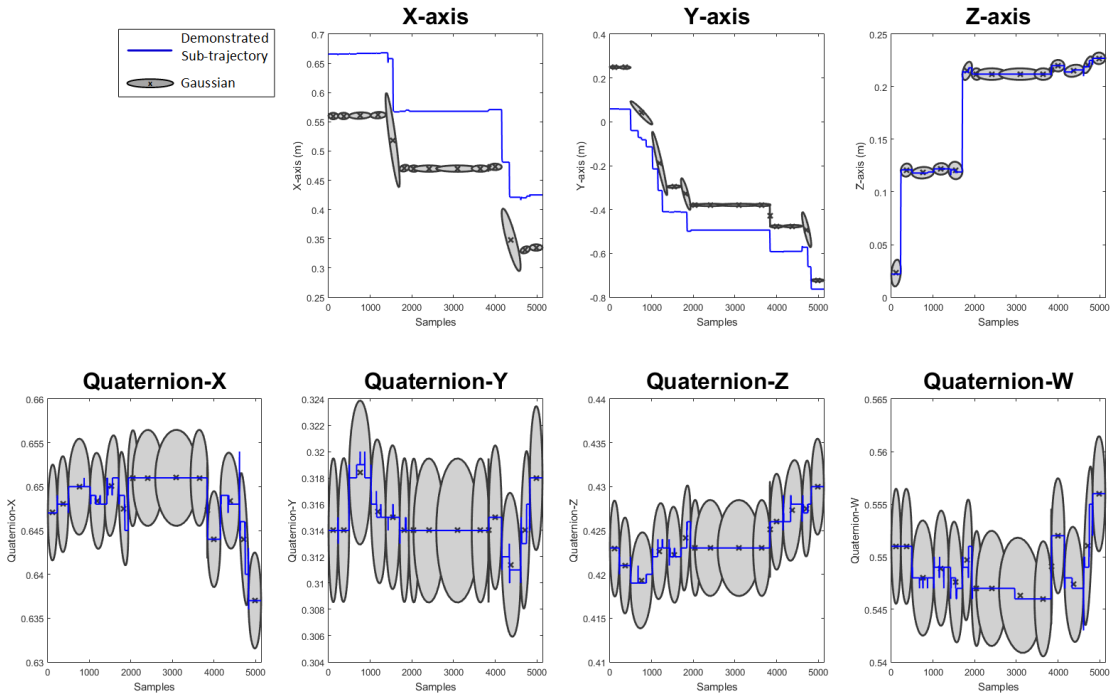


Figure 96: Assistive Task - The modification of the GMM for the second sub-trajectory – Trial 2

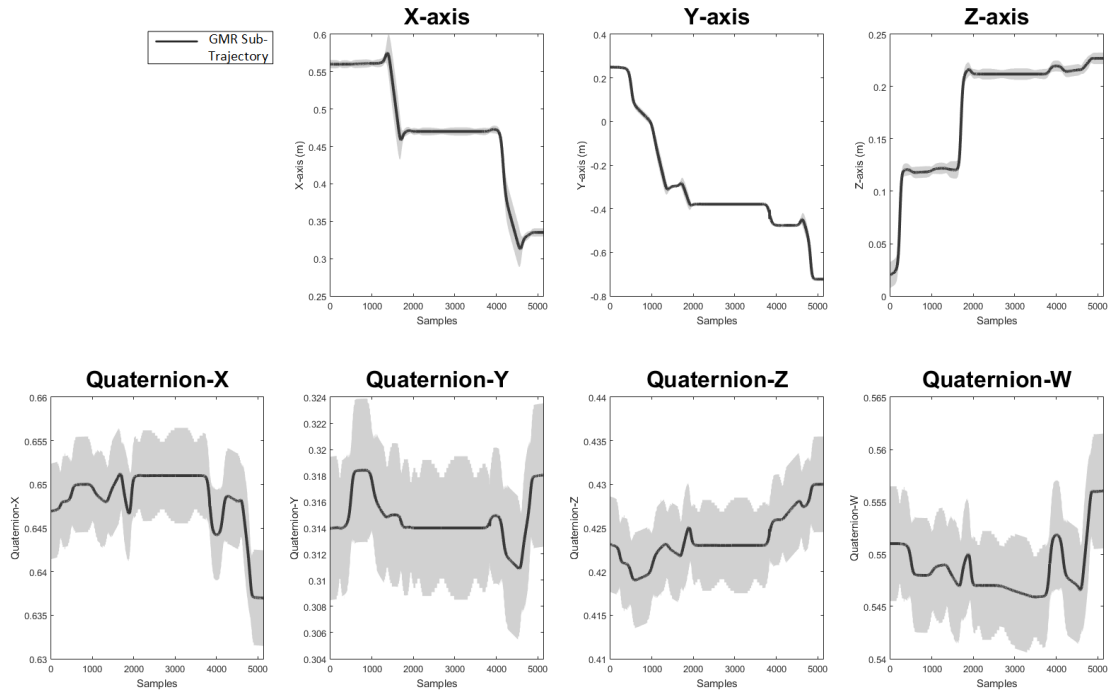


Figure 97: Assistive Task - The second GMR sub-trajectory produced by the m-GMM – Trial 2

- **Trial 3:** The output of m-GMM/GMR for the first sub-trajectory

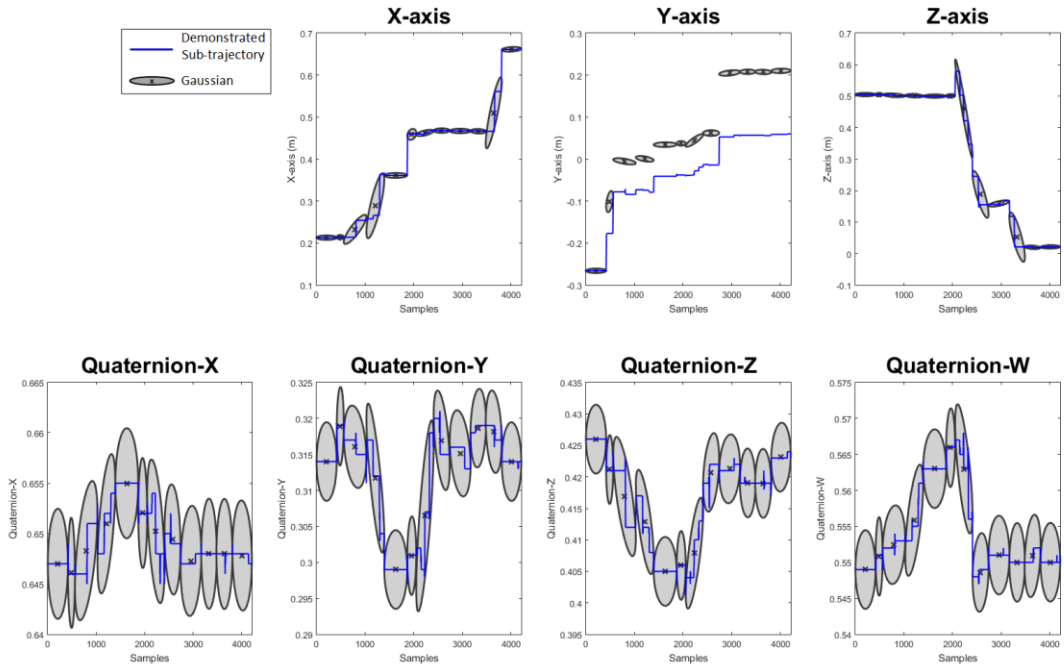


Figure 98: Assistive Task - The modification of the GMM for the first sub-trajectory – Trial 3

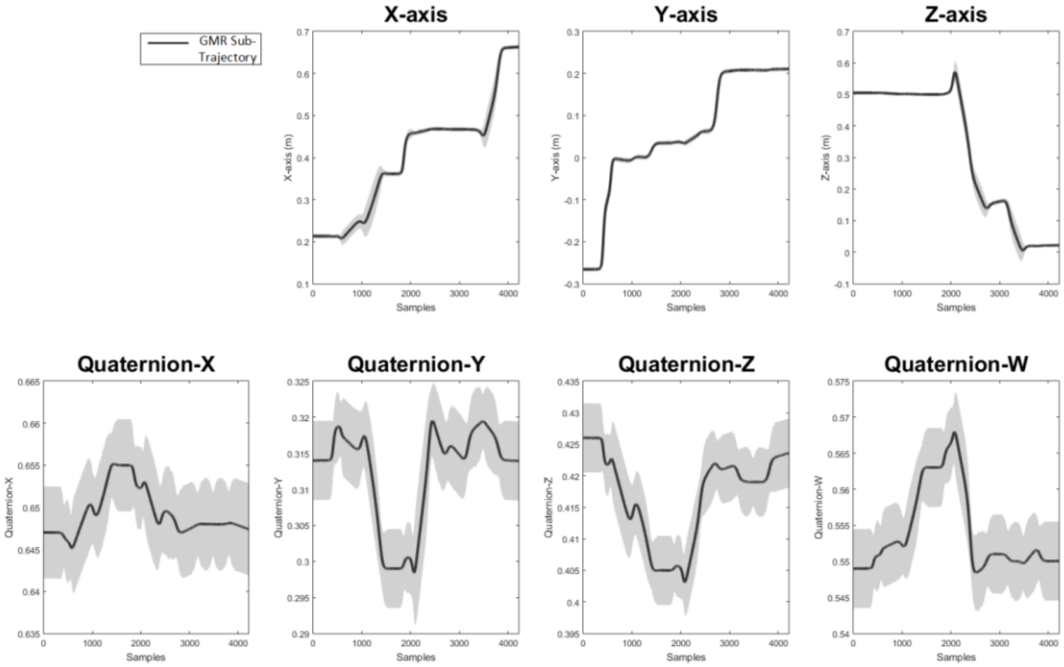


Figure 99: Assistive Task - The first GMR sub-trajectory produced by the m-GMM – Trial 3

- **Trial 3:** The output of m-GMM/GMR for the second sub-trajectory

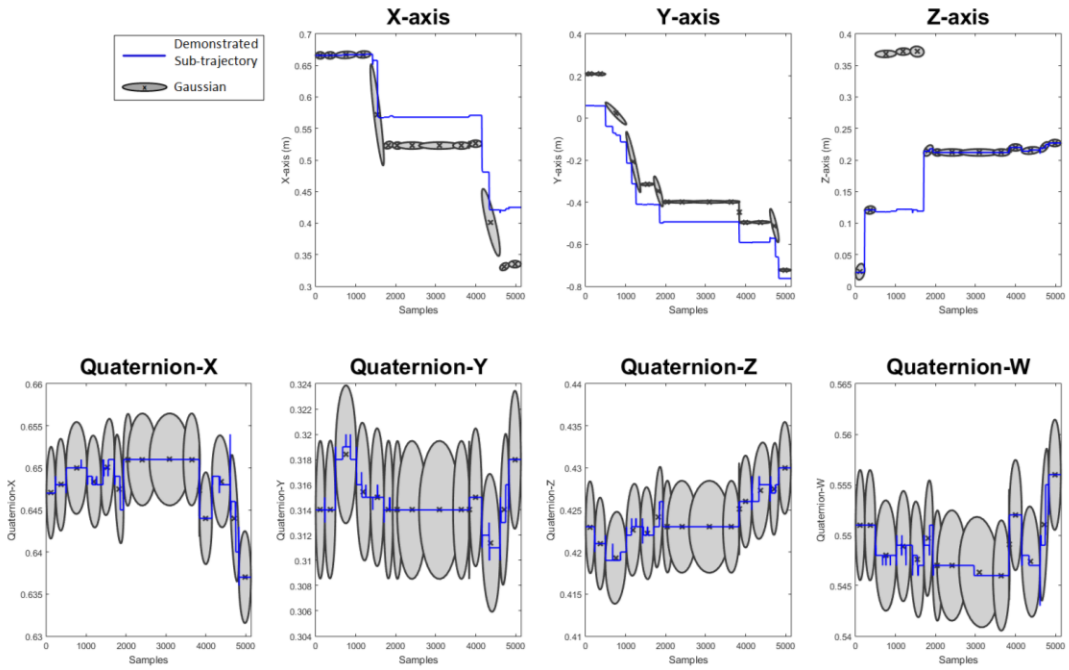


Figure 100: Assistive Task - The modification of the GMM for the second sub-trajectory – Trial 3

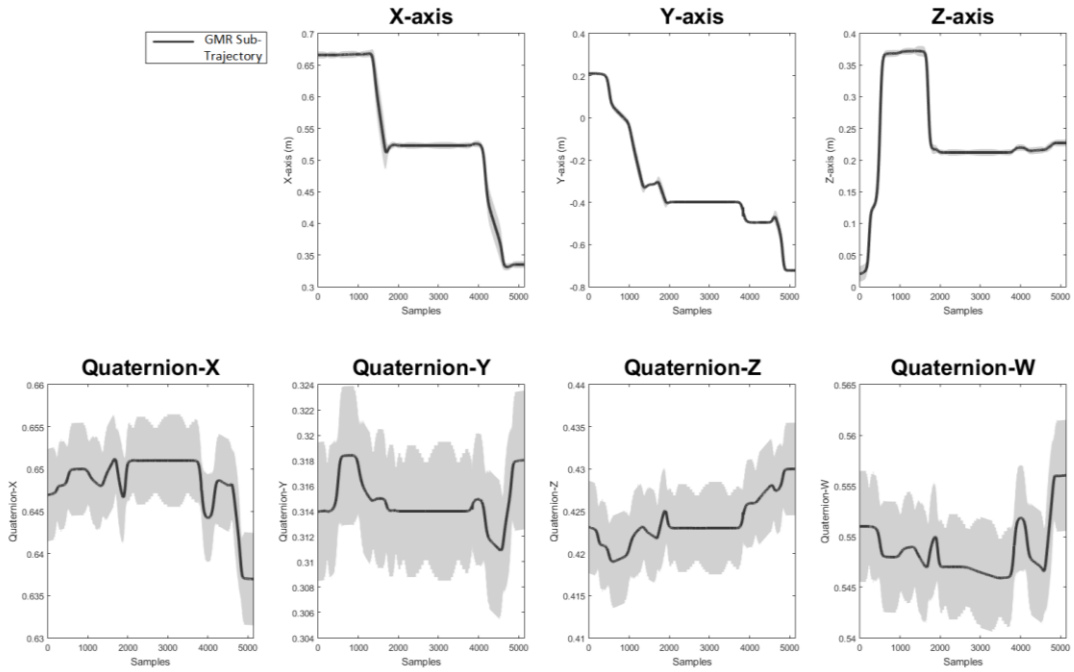


Figure 101: Assistive Task - The second GMR sub-trajectory produced by the m-GMM – Trial 3