# DLR-IB-RM-OP-2021-67

## Recognition and Reproduction of Force-based Robot Skills via Learning from Demonstration

**Masterarbeit**

Vamsi Krishna Origanti

Deutsches Zentrum
DLR für Luft- und Raumfahrt



This thesis was submitted to the Institute of Mechanism Theory, Machine Dynamics and Robotics

# Recognition and Reproduction of Force-based Robot Skills via Learning from Demonstration

Master Thesis

*by*:

Vamsi Krishna Origanti

Student number: 394 822

*supervised by:*

Dipl.-Ing. Dipl.-Wirt.Ing. Michael Lorenz,
Sascha Weil M.Sc.,
RWTH Aachen University

Thomas Eiband M.Sc.,
Deutsches Zentrum für Luft- und Raumfahrt

*Examiner*:

Univ.-Prof. Dr.-Ing. Dr. h. c. Burkhard Corves

Prof. Dr.-Ing. Mathias Hüsing

Aachen, 14 April 2021

## Master Thesis

by Vamsi Krishna Origanti

Student number: 394 822

**Recognition and Reproduction of Force-based Robot Skills via Learning from Demonstration**

With an increase in the demand for complex tasks in industrial applications such as assembly tasks, industries foresee demand for robots with unique abilities. For example, there is a requirement for more customizable robots and adapt to dynamic changes in the environment. Interactive tasks are such complex tasks where robots should adapt to changes to interact with the environment. Such tasks are also called contact-based tasks or compliant tasks. The robot performing compliant tasks should be able to possess the skills associated not only with kinematic movements but also force profiles and corresponding control schemes. Programming a robot to execute contact-based tasks can be time-consuming, where usually expert knowledge is required. Learning from Demonstration (LfD) provides an intuitive way to deal with such complex tasks with minimal programming effort. As such, demonstrations might not always lead to an efficient behavior, but the intent of the user can be recognized based on the motion and force data. The goal is to extract the real intent of the user, which is to exhibit contact-based skills that are adaptable to changes and not simply replay the demonstration.
Skill templates need to be developed that are parameterized by the demonstration to reproduce the skill efficiently. This thesis aims to provide a methodology to identify such skill templates for commonly used industrial tasks such as slide, e.g., to surface polishing, touch to slight contact to identify the objects or constraints, press to apply forces on to the environment for e.g., pressing a button and contouring to perform tasks such as deburring of manufactured parts. This thesis presents methods employed to identify and extract these features required to represent a skill template capable of reproducing desired skills. Additionally, a control strategy is derived for hybrid position-force control to reproduce the skills from skill templates. The methodologies employed are evaluated, and the implications are inferred by reproducing contact-based skills under PyBullet simulation environment configured with an LWR-IV robot.

Supervisor: Dipl.-Ing. Dipl.-Wirt.Ing. Michael Lorenz,
Sascha Weil M.Sc.,
RWTH Aachen University
Thomas Eiband M.Sc.,

Deutsches Zentrum für Luft- und Raumfahrt

## Eidesstattliche Versicherung

Vamsi Krishna Origanti Matrikel-Nummer: 394 822

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Master Thesis mit dem Titel

**Recognition and Reproduction of Force-based Robot Skills via Learning from Demonstration**

selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Aachen, 14 April 2021

Vamsi Krishna Origanti

**Belehrung:**

**§ 156 StGB: Falsche Versicherung an Eides Statt**

Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

**§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt**

(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.

(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

Aachen, 14 April 2021

Vamsi Krishna Origanti

The present translation is for your convenience only.
Only the German version is legally binding.

**Statutory Declaration in Lieu of an Oath**

Vamsi Krishna Origanti Student number: 394 822

I hereby declare in lieu of an oath that I have completed the present Master Thesis entitled

**Recognition and Reproduction of Force-based Robot Skills via Learning from Demonstration**

independently and without illegitimate assistance from third parties. I have used no other than the specified sources and aids. In case that the thesis is additionally submitted in an electronic format, I declare that the written and electronic versions are fully identical. The thesis has not been submitted to any examination body in this, or similar, form.

Aachen, 14 April 2021 Vamsi Krishna Origanti

**Official Notification:**

**Para. 156 StGB (German Criminal Code): False Statutory Declarations**

Whosoever before a public authority competent to administer statutory declarations falsely makes such a declaration or falsely testifies while referring to such a declaration shall be liable to imprisonment not exceeding three years or a fine.

**Para. 161 StGB (German Criminal Code): False Statutory Declarations Due to Negligence**

(1)If a person commits one of the offences listed in sections 154 to 156 negligently the penalty shall be imprisonment not exceeding one year or a fine.
(2) The offender shall be exempt from liability if he or she corrects their false testimony in time. The provisions of section 158 (2) and (3) shall apply accordingly. I have read and understood the above official notification: :

Aachen, 14 April 2021 Vamsi Krishna Origanti

## Contents

## Formula symbols and indices

### Lower case latin letters as formula symbols

| | | |
|---|---|---|
| $a$ | $\frac{\mathrm{m}}{\mathrm{s}^2}$ | acceleration |
| $t$ | s | time |
| $x$ | m | State variable or input variable |
| $x_0$ | m | Initial state variable |
| $\dot{x}$ | $\frac{\mathrm{m}}{\mathrm{s}}$ | Derivative of state variable or input variable |
| $g$ | m | Goal state variable or gravity compensation |
| $s$ | – | Phase variable |
| $w_i$ | – | Weights |
| $f$ | – | Function or a force-torque vector |
| $q$ | – | Quaternion or joint angles |
| $p$ | – | Translation vector |

**Upper case latin letters as formula symbols**

| | | |
|---|---|---|
| $K_c$ | – | Stiffness Matrix |
| $D_x$ | – | Damping Matrix |
| $J$ | – | Jacobian Matrix |
| $K_p$ | – | parameter of PI force controller |
| $K_i$ | – | parameter of PI force controller |
| $S$ | – | Selection Matrix |
| $\tilde{S}$ | – | compliment of Selection Matrix |
| $T$ | – | Homogeneous Transformation Matrix |
| $R$ | – | Rotation Matrix |
| $\mathcal{N}$ | – | Normal Distribution |

**Lower case greek letters as formula symbols**

| | | |
|---|---|---|
| $\alpha$ | $-$ | DMP model parameter |
| $\beta$ | $-$ | DMP model parameter |
| $\delta$ | $-$ | quaternion difference |
| $\omega$ | $\frac{\text{rad}}{\text{s}}$ | angular velocity |
| $\dot{\omega}$ | $\text{rad}/\text{s}^2$ | angular acceleration |
| $\tau_{ic}$ | $\text{N}\,\text{m}$ | Joint Torques from Impedance Controller |
| $\tau_{fc}$ | $\text{N}\,\text{m}$ | Joint Torques from Force Controller |
| $\tau$ | $\text{N}\,\text{m}$ | Joint Torques from Hybrid Controller |

**Upper case greek letters as formula symbols**

| | | |
|---|---|---|
| $\Omega$ | – | Parameter of Modified Hybrid Controller |
| $\tilde{\Omega}$ | – | Parameter of Modified Hybrid Controller |
| $\mu$ | – | Mean |
| $\Sigma$ | – | Covariance |
| $\Psi_i$ | – | Radial Basis Function |

**List of abbreviations**

| | |
|---|---|
| **LfD** | Learning from Demonstration |
| **PbD** | Programming by Demonstration |
| **MP** | Movement Primitive |
| **DMP** | Dynamic Movement Primitive |
| **CF** | Compliant Frame |
| **EF** | Endeffector Frame |
| **BF** | Base Frame |
| **LWR** | Linear Weighted Regression |
| **GPR** | Gaussian Process Regression |
| **GMR** | Gaussian Mixture Regression |
| **GMM** | Gaussian Mixture Models |

**HMM** Hidden Markov Models

**SEDS** Stable Estimator for Dynamic Systems

**DLR LWR** Deutsche Zentrum für Luft- und Raumfahrt's Light Weight Robot

**DOF** Degree of Freedom

**GUI** Graphical User Interface

**API** Application Programming Interface

**LN** Links and Nodes

**URDF** Unified Robot Description Format

**SDF** Spatial Data File

**DTW** Dynamic Time Warping

**PCA** Principal Component Analysis

**DND** Directional Normal Distribution

# 1. Introduction

In recent years, the growth of usage of robots is increasing very rapidly, especially in industries. The world is witnessing the advantages of robotization in a factory setting, and demand for robots increased and spread across other sectors such as logistics, service robotics, medical robotics, etc. However, despite the growth and demand, robots are most commonly preprogrammed and hard to realize automatic skill learning and adapt to environmental changes. A skill is a predefined robot behavior parameterized by demonstrations [PNA16][ESL19]. Recently, a lot of consideration towards skill learning for robots has increased. Learning contact-based skills is further challenging as it requires learning and adapting the interaction behavior towards the environment. This thesis aims to develop a framework to learn predefined skills that deal with contact-based or compliant tasks. The intention to learn such predefined skills is to build a skill library for the most commonly used skills in industries. This library can be used to frame task-level programming by grouping these skills.

Robots that are involved in tasks where they undergo physical interactions with the environment, an active force application also needs to be considered. Such complex tasks can be defined as Force-based tasks, also known as Compliant tasks. The robots performing compliant tasks should be able to possess the skills that associate not only with kinematic movements but also force profiles and corresponding control schemes. However, it gets more challenging to preprogram robots for all circumstances. Force-based tasks are relatively hard to program compared to only kinematic tasks.

This thesis work employed Learning from Demonstration (LfD) techniques that provide scope for an intuitive way to deal with such complex tasks with minimal programming effort. The main idea is to develop a methodology that robots learn skills from the demonstrated data, ideally from a single demonstration that allows robots to reproduce the skill and adapt to dynamic changes in the environment. On this note, the main focus of this thesis is to develop a methodology to learn contact-based skills and be able to reproduce by using the LfD technique. This chapter discusses the background, including the goals, objectives, and contributions of the thesis work.

### 1.1. Problem Statement

Due to the high demand for contact-based applications using LfD, compliance control strategies were developed along with position control for replicating the demonstrated tasks. Learning of force-based tasks from the demonstration were addressed well in [CH18] [KGS15] [SMK15]. The state-of-the-art methods in contact-based tasks mentioned in [CH18] addressed the limitations of conventional methods when forces are time-varying and multidimensional. However, these approaches are formulated as non-specific to each skill and more towards a generalized way of dealing with contact-based problems. Even though this generalized behavior can deal with the contact-based learning problem, some tasks such as peg-in-hole applications as mentioned in [BD96] are highly complex and demand more specific approaches, which are different from other tasks. Hence, an LfD framework for skill-based learning is proposed here to address this issue. The main motivation behind skill-based learning is to provide an intuitive way of programming robots in industries by providing a set of skill libraries. This reduces the programming effort and does not require expert knowledge to program complex industrial assembly tasks. Bøgh et al. in [BNP12] claim that the skills are the foundation for task-level programming of robots, which can provide huge customization and flexibility. This idea of a skill-based learning framework for contact-based tasks opens up challenges like

1. How to define and identify such skills?
2. What are the features to be extracted and how to perform skill parameterization?
3. How to implement the above mentioned compliant control methods to skills?
4. How to choose the control sequence specific to skill?
5. How to embed complex skill-specific algorithms, for example, dedicated algorithms for peg-in-hole applications?.

This project's scope, methodology, and work plan to address these challenges are explained further in subsequent sections.

## 1.2. Goals and Objectives

This thesis aims to develop a Skill-based learning framework for robots to perform contact-based tasks that are learned by Kinesthetic teaching. The motivation is to develop a framework for predefined skills, which are commonly used in industries. And to propose a user-friendly way to teach complex compliant tasks intuitively without any prior knowledge and with minimal demonstrations, ideally from a single demonstration. One possible way to achieve this goal is by employing methods that are engineered more specifically for each skill rather than adopting a generic way for all skills for learning and extracting parameters or features. Because the skill behavior would vary among skills and such generic approaches would fail to satisfy skill specific requirements.

As detailed in the problem definition, this idea of developing skill-based learning raises challenges on how to realize the working methodology of this thesis. Considering those challenges, the goal of the thesis is to develop a framework for skill-based learning that can be achieved in five steps 1)Skill Definition, 2)Skill Identification, 3)Skill Parameterization, 4)Controller Implementation, and 5)Skill Reproduction.

Defining and identifying Skills are not in the scope of this thesis work. Existing works on defining and identifying skills from demonstration data are used as pre-steps. After identifying the skill, the major works can be described in three tasks:

1. Define a meaningful set of contact primitives for skills and extract their parameters such that they allow the parameterization of a controller.
2. Define and implement the necessary controllers which can optimize the reproduction.
3. Execute the sequenced contact primitives and evaluate their optimization and generalization capabilities compared to a simple replay of the demonstration.

The goals and main objectives are illustrated in Fig.1.1.

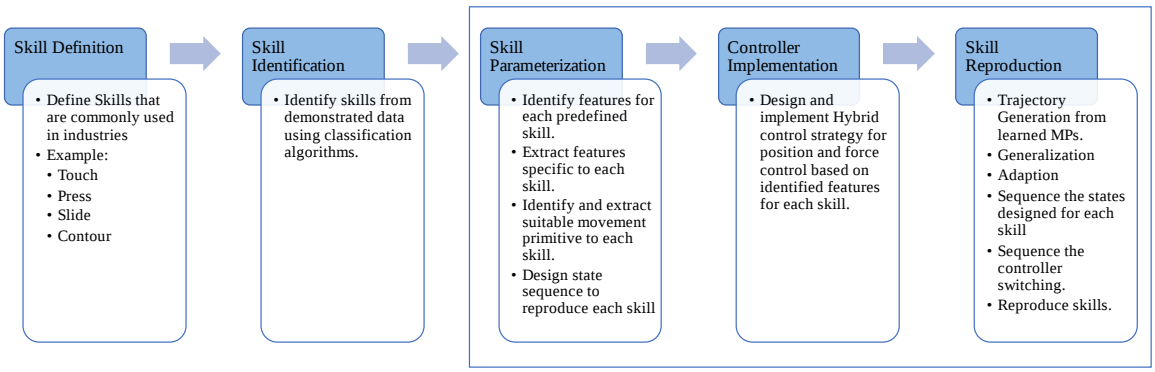



**Figure 1.1.:** Project Goals and Objectives

## 1.3. Contributions

Major contributions of this thesis to develop a framework for Skill-based learning of contact-based tasks are enumerated as below:

1. Proposes a methodology for *Skill parameterization* of each predefined skills. Here, as a part of the thesis work, four important preliminary skills: Touch, Press, Slide and Contour are chosen. This methodology can be extended to other skills with minimal modifications.

2. Provides a theoretical analysis of skill learning and *Feature Identification* of important features that are enough to represent a contact-based skill. These identified features are the foundation of the thesis work to learn a skill that can reproduce the desired task.

3. Proposes a theoretical formalism for the *Feature Extraction* of the identified features for each skill. Identified features are extracted methodologically that vary between each skill.

4. Establishes *motion encoding by learning Dynamic Movement Primitives (DMP).* Trajectory generation from learned DMPs is based on Linear Weighted Regression (LWR). DMPs help to adapt to the changes in the initial and goal points. This feature of DMPs is being used for devising generalization and adaption capabilities for each skill.

5. Provides a *control strategy for a hybrid position-force controller* that also facilitates rigid transformation between frames. This control strategy helps us to choose control parameters and switch sequence between position and force control for each dimension, most importantly, in the task frame. This is an important criterion required to reproduce skills.

6. Setup of *simulation environment* for a robotic system using PyBullet python package, which offers simulations that includes forces and torques. PyBullet is interfaced with the hybrid controller implemented in MATLAB Simulink to control the robot loaded into the environment. The simulation environment configured can further be used for a variety of robotic systems, not limiting to the robot model used only in this thesis work.

# 2. Background

This section provides a brief background relevant to skill learning and covers the basic introduction of some important topics such as Learning from Demonstration, Movement Primitives.

## 2.1. Learning from Demonstration

Learning from Demonstration(LfD), also known as Programming by Demonstration (PbD), is a disruptive methodology that offers robots to learn manipulation behaviors from observing the motions executed by human demonstrators. The aim is to teach robots in a natural and intuitive way without any prior programming knowledge. In recent years, LfD has consequently received a great deal of attention in the field of Robotics and the present-day goal of LfD is to learn a policy or control program of a task that is robust to noise, initial conditions, and generalization capability to be able to handle variations in tasks or environments.[BG13]

The main principle of robot LfD is that users can teach tasks to the robot without programming, which can be carried out in three steps: Demonstration, Skill Acquisition, and Execution as shown in Fig.2.1.

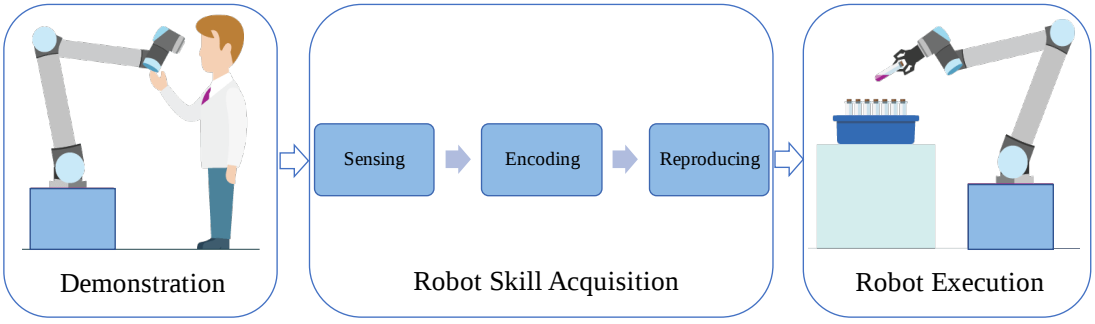



**Figure 2.1.:** Learning from Demonstration

### 2.1.1. Demonstration

Human demonstrations are captured to learn the tasks or skills, which include different strategies like Kinesthetic Demonstration, Motion-Sensor Demonstration, and Teleoperated Demonstration [ZH18].

In Kinesthetic Demonstration, the robot arm is moved physically by a demonstrator in Zero Gravity mode, or back drive mode of robots [SMK15]. During the demonstration, the movements are directly recorded with the position, orientation, and wrench data, which are used further to process and encode movements. The main advantage of Kinesthetic demonstration is that the data can be recorded directly without any modification or transformation like pose estimation, correspondence mapping between human pose and robot, etc., as required in other strategies [ZH18].

The Motion-Sensor Demonstration requires a motion capture system that often employs visual tracking of markers that are positioned at the human body to record demonstrated movements. Imitation learning is carried out by mimicking the recorded motions by a robot using different types of computer vision strategies, which are more often used in the case of humanoid robots. This methodology for capturing demonstration comprises three components: human motion measurement, motion mapping, and motion control [CL18][ZH18].

Teleoperated Demonstration is similar to Kinesthetic Demonstration, wherein the direct mapping of human motions to the robot is possible. During the teleoperation demonstration, the teacher usually manipulates the robot through an input device, standing far away from the robot. Considering assembly tasks, when a human demonstrator performs the assembly motions, the movement information of the human arm is fed into a real-time tracking system so that the robot can mimic the movements of the demonstrator. Teleoperation has the benefit of setting up an effective correspondence and activity technique among humans and robots. It has been applied in different applications as listed in [ZH18] including remote control of a mobile robotic assistant, performing an assembly task, performing a spatial-positioning task, demonstrating grasp preshapes to the robot, transmitting both dynamic, communicative information between demonstrator and robot on a collaborative task, picking and moving tasks.

### 2.1.2. Robot Skill Acquisition

The demonstrated task is preprocessed to be used for learning tasks and skill acquisition. The main goal of this step is to meaningfully interpret and learn the movements from the demonstration data that can be used for further processing to reproduce the task. Skill Acquisition consists of three main phases as shown in Fig.2.2 [CL18] [XHL19].

In the Sensing Phase, demonstrated motion trajectories including position, forces, and actions are interpreted and preprocessed. This step overlaps with the demonstration, wherein the data is recorded while performing a demonstration in real-time. The recorded

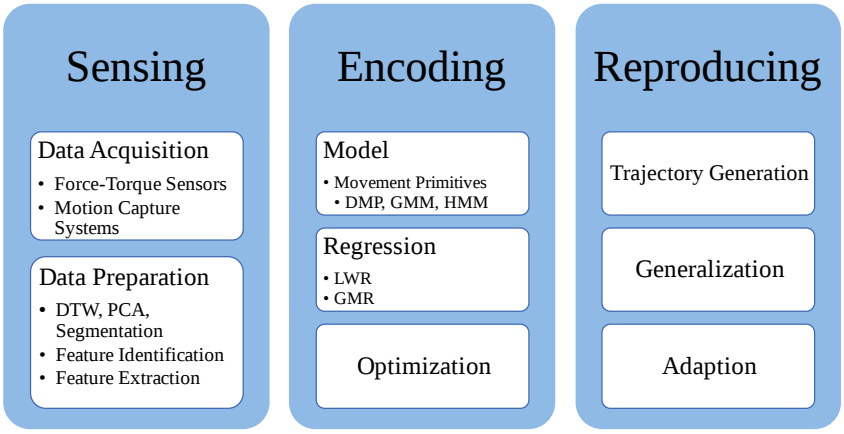



**Figure 2.2.:** Learning Process

data is processed to frame a meaningful set of parameters that can be used in learning, for instance, dimensionality reduction, feature extraction, and data fusion. Sensing is performed differently based on the selected Demonstration strategy. In the case of Kinesthetic Demonstration, the sensing is performed by tactile and position sensors mounted on the robot. In contrast, in Motion-Sensor Demonstration, sensing is performed by kinematic pose estimation using optical sensors and vision tools [ZH18] [CL18] [XHL19]. For a better understanding of data, data preparation and preprocessing steps are carried out before encoding the movements. Data preprocessing such as Dynamic Time Warping(DTW) for aligning trajectories, Principal Component Analysis (PCA) for dimensionality reduction, segmentation for splitting trajectories simplify encoding for complex tasks. Features are identified and extracted that are required in the sensing phase after acquiring data from demonstrations for further processing during learning and reproducing. These features are often referred to as skill templates that enhance the interpretation of data more specific to the task and allows to learn complex skills with minimum computational efforts [XHL19][LKY13].

Encoding is a process of mapping the movement trajectories and motions obtained by human demonstrations to a meaningful mathematical model, which is learned to regenerate desired motion trajectories. The mathematical model of encoded motions is also referred to as Movement Primitives. In contrast to simply saving demonstrated trajectory data, encoding motions in the form of movement primitives offers a salient feature that is generalization capability to adapt to the dynamic changes for new goals and initial points. MPs can also be represented as building blocks that can be organized in series and parallel to create more complex behavior [CL18]. Different types of such strategies and Movement Primitives as a representation of motions under learning scenario is explained in detail in [CL18]. The type of MP most often used is Dynamic Movement Primitives (DMP) [INH13]. Schaal et al further explored DMP under learning scenarios in [SPN05]

[SPN03]. DMP is a nonlinear dynamic model to encode the trajectories of the demonstrated task. A second-order differential equation is used to encode the desired MPs as a typical spring-damper system with nonlinear forcing terms [INH13] [CL18]. Other Movement Primitive representations are Gaussian Mixture Models (GMM), Gaussian Mixter Regression (GMR), and Hidden Markov Model (HMM) are further explained in detail in the following section. In [CL18] a detailed explanation and comparison of these models are provided.

Reproducing the encoded skills is quite challenging and it is required to reproduce each skill such that it is robust to perturbations. Additionally, each skill should adapt to the task requirements. During the reproducing phase, optimal trajectory generation, generalization, and motion reconstruction based on task requirements are carried out. For instance, MPs are capable of generating trajectories with the change in goal points. This change in goal points is adapted to the environment based on skill or task requirements. The optimized and generated motion adapted to the task or skill is then utilized to perform the desired task in the robot execution level.

### 2.1.3. Robot Execution

Different learning control strategies as a problem of estimating control policies for discrete dynamical systems are described comprehensively in [CL18]. The reference trajectories generated in the reproducing phase of skill acquisition are tracked and controlled with a controller in order to execute the learned task from the demonstration.

Impedance control which is based on a spring-damper system, is a commonly used control strategy in LfD methods for controlling learned positions from MPs. However, for compliant tasks, where the influence of forces is considered, simple impedance control is not sufficient. Though the impedance controller can apply forces onto the environment, additionally, a feedback force controller is required to track the forces extracted from skill acquisition when the robot is in contact with the environment. Early works on embedding force control along with impedance control as in [LKY13] [SMK15] gave significant results. Conkey et al. in [CH18] proposed a robust way of dealing with hybrid position-force control. Control parameters should be extracted and engineered based on the task specification and demonstrated data. Also, a control sequence needs to be chosen for MPs such that the robot will execute the reproduction of demonstrated contact-based skills.

## 2.2. Movement Primitives

Humans are capable of interpreting and reproducing object manipulation and articulated tasks that are underlying with motor skills. However, in robots, tasks are usually performed by executing precise movements that are preprogrammed in a point-to-point manner. There are limitations to program complex tasks in robots that humans are able to perform. Therefore, there is a requirement for a symbolic and systematic representation of movements. Such complex movements underlying motor skills can be obtained by grouping elementary motion representations called Movement Primitives [SL13]. MPs are mathematical representations of motion, where movements are encoded into a mathematical model by learning from the motion data such as position, orientation, and forces profiles recorded during the demonstration. MPs are considered basic building blocks or elementary movements that can be combined sequentially to form a complex motion.

Conventionally, the encoding of a movement has been done by utilizing splines, and Bézier curve [Ude93] [HAK03]. However, these methods have limitations to learn skills, limited capabilities for generalization and adaption and have no control on other important variables like velocity and acceleration. In recent years, the focus has shifted to statistical models like GMM, HMM, GP etc. [CFS10] [MGH09] [CGB07] [NP08] and dynamical system models [HGC08] [KB10] [KB11] like SEDS and DMP to encode human demonstrations.

In the context of LfD, MPs are the representation of movements that are learned from demonstration and should also possess the ability to generate trajectories that support generalization and adaption to novel situations. Hence, trajectory generation from the model can be seen as a regression problem that is able to generate motion considering new initial and goal points. Many algorithms emerged to encode movements, among which DMM, GMM, HMM, and their variants have gained considerable attention under the LfD context to represent motions. In this chapter, the role of MPs is explained in LfD, skill learning, and introduces different types of MPs aforementioned.

Learning a movement primitive is to find a control policy that is specific to a task. For instance, in DMPs that represents movements, a control policy is derived on a nonlinear dynamic systems guarantees reaching desired goal position. Consider a dynamical system represented in the form of differential equation [SPN03]:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \alpha, t) \tag{2.1}$$

Once the policy is learned for a given task, which represents the kinematic planning of the motion, trajectories can be generated directly to existing states or novel situations

with change in initial and goal states. In simple words, the policy learned depicts the movement between any two points given. The output of the MPs is subsequently used as reference commands of an appropriate controller in each dimension, as shown depicted in Fig.2.3.

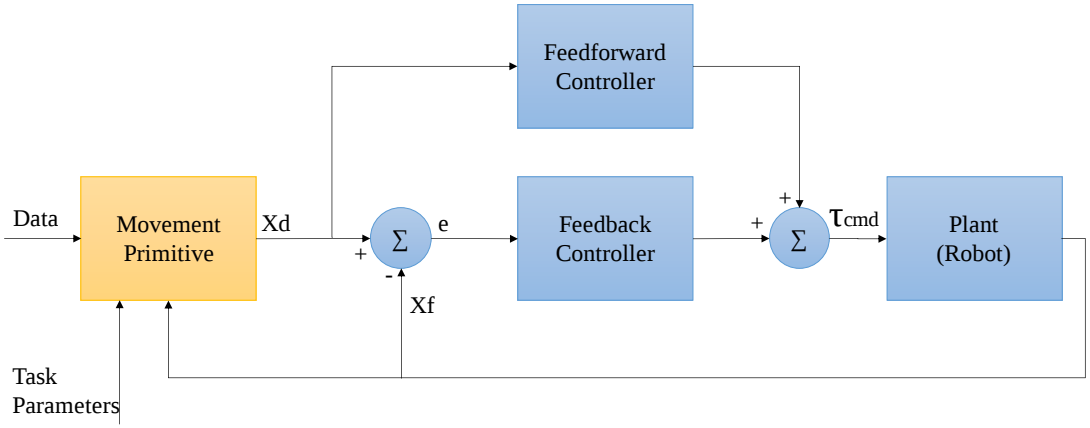



**Figure 2.3.:** Movement Primitive with Controller, Source:[SPN03]

As stated earlier, there exist different ways to represent motion as MP. Following are some of the well-known methods often used to represent motion as Movement Primitives in LfD and skill learning scenarios.

### 2.2.1. Locally weighted Regression

Locally Weighted Regression Learning (LWR) is a non-parametric regression method [AMS97]. The regression problem to generate output data completely relies on data points that are close to the input values. Regression can be achieved by extending the weighted least squares performed on a data set with a superposition of basis function, most commonly Radial basis function (RBF). It targets to fit a nonlinear problem by splitting and solving locally by linear regression. Further explanation about theoretical formulation is given in A.1.1

Sample trajectory learning using LWR is shown in Fig.2.4 with different degrees of a polynomial. The basis functions were chosen accordingly. Detailed explanation and examples are provided in the [CL18].

### 2.2.2. Gaussian Process Regression

In Gaussian Process Regression (GPR), considering the regression problem in the form of $\boldsymbol{y} = f(\boldsymbol{x}) + \eta$, where $f$ is an unknown function that should be estimated and $\eta$ is

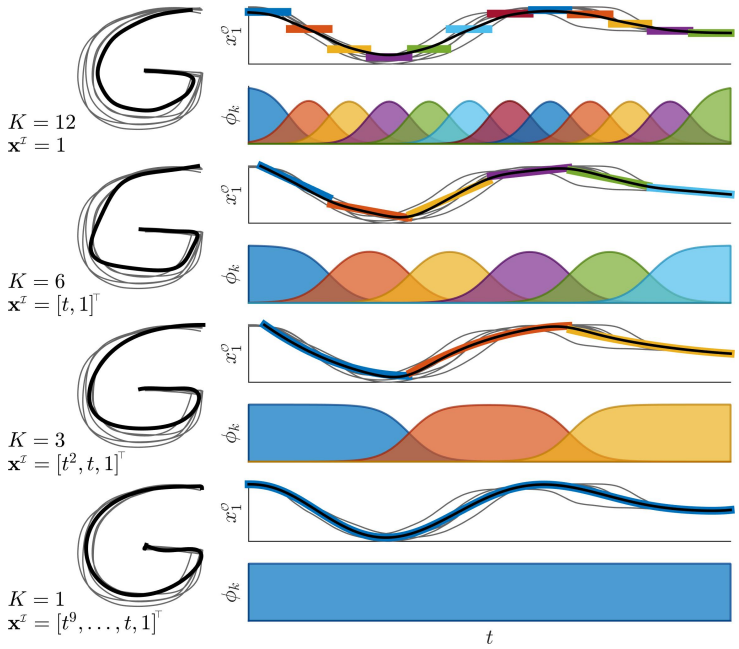



**Figure 2.4.:** Sample Trajectory learning using LWR, Source:[CL18]

an additional noise. Consider a data set obtained from the demonstration consisting of input and output pair, and the goal is to evaluate the unknown function $f$ using probabilistic distribution. The input and output mapping learned is called Gaussian Process Regression. Here, each observation in the data can be considered as samples drawn from a multivariate Gaussian Distribution. Which means, the unknown function $f(\boldsymbol{x})$ that is associated to the input $X$ is a sample drawn out of a multivariate gaussian, for example $\mathcal{N}(\boldsymbol{\mu}(\boldsymbol{x}), \boldsymbol{K})$, where $\boldsymbol{K}$ is a covariance matrix also called as Gramm matrix that is defined using kernel function $\boldsymbol{k}(x_1, x_2)$ that provides covariance between two samples of dataset.[CL18]. Further details of the mathematical formulation is provided in A.1.2.

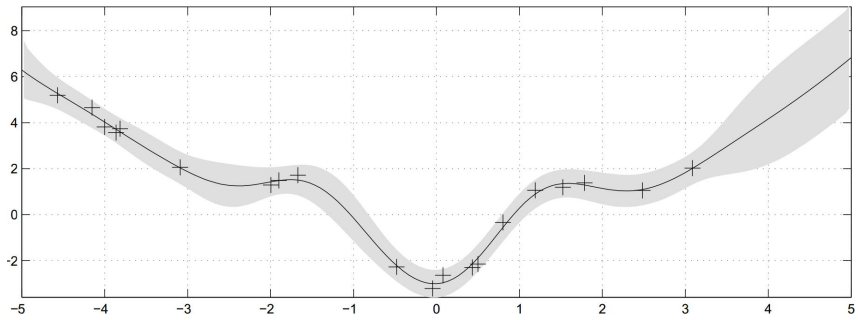



**Figure 2.5.:** Sample Trajectory learning using GP, Source:[RW]

### 2.2.3. Gaussian Mixture Regression

Gaussian Mixture Regression (GMR) is a regression method to learn movement primitives in demonstration data that is encoded as a probabilistic model on the joint proba-

bility density of data in the form of Gaussian Mixture Model (GMM) or Hidden Markov Model (HMM)[CGB07]. Estimation of the model is performed by using the Expectation-Maximization (EM) procedure. The regression function is then computed from the joint probability density model learned from data. GMR is also a popular probabilistic method for movement representation, which is based on linear transformations and conditional properties of multivariate normal distribution [CL18]. The computational complexity of GMR is smaller as compared to GPR as it does not estimate the regression function directly but computes the regression function from the model. Another advantage is that the computation time is independent of the sample space of data.

The data to be learned can be multidimensional, and the expectation can be computed by considering multivariate distribution. If multiple demonstrations are recorded and expected to learn from them, it is hard to define a joint probability over the entire dataset. To overcome this issue, often Dynamic Time Warping as a preprocessing step is considered before modeling data into GMM [MGH09]. Overview of GMR process with GMM or HMM can be shown in Fig.2.6

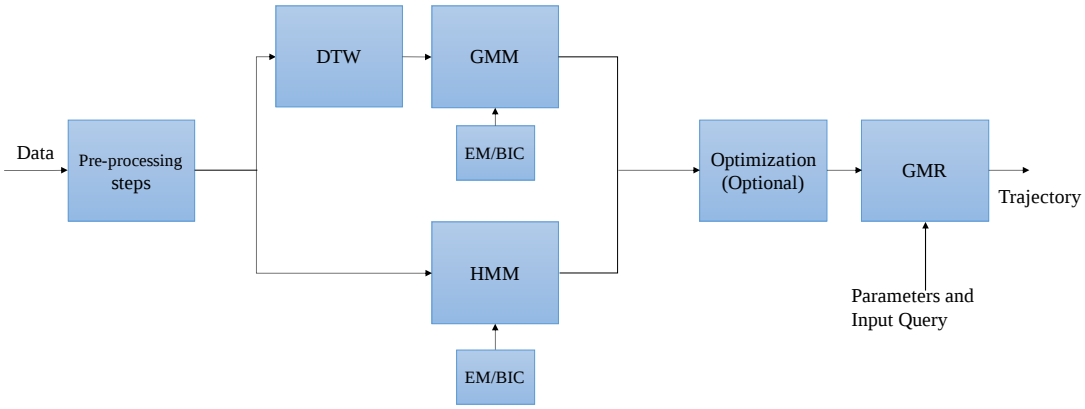



**Figure 2.6.:** Overview of GMR process

Figure 2.7 shows the formulation of GMR using the GMM model in 1-Dimension. GMR can cover multiple regression methods from simple linear regression to multivariate and kernel-based regression. [CL18]. Refer A.1.3.1 for the mathematical theory behind GMM.

Another way to model data to use with GPR is Hidden Markov Model (HMM). Similar to GMM, HMM also comprises a mixture of Gaussians but is represented as states. Along with prior, mean and covariance matrix, model comprises a transition matrix that defines transition probability between the states. For further explanation on the theory of HMM, refer A.1.3.2.

Nevertheless, both GMM and HMM would require to specify the number of Gaussians to be used to model the data. Bayesian Information Criteria (BIC) is one of the com-

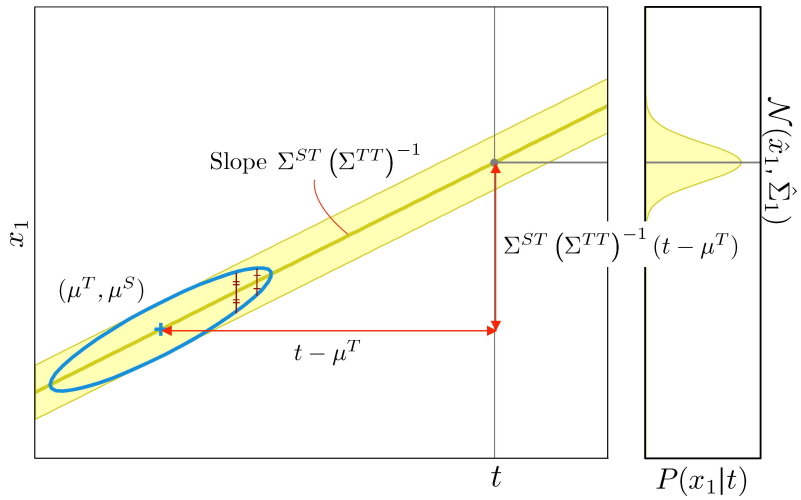



**Figure 2.7.:** Gaussian Mixture Regression on 1D trajectory with GMM model, Source:[CL18]

mon approaches that suggest the optimal number of Gaussians that can be utilized [CGB07][MGH09].

After encoding the dataset into either of the models, GMM or HMM, GMR is used to obtain the target output. In this context, the target trajectory represents the movements for a given input query. The Gaussians that establish the model can be considered as attractors that offer a minimized representation of the movements. For GMMs, time is given as input [CGB07][MGH09], while for HMMs it is the current state as explained in [CFS10]. The result is determined by linear combinations of a mixture of the Gaussians. For GMM, the weighting of each Gaussian depends on their likelihood. For HMM, it is based on transition probability that is based on the previous state.

### 2.2.4. Movement Primitive learning based on Dynamical Systems

Though GMR with GMM or HMM show considerable results and are used in imitation learning, these methods cannot generalize to novel situations. For instance, when a starting point is too far from the starting point of the demonstration, the final trajectory generated may not follow the intended path shown in Fig.2.8a. To overcome this issue, [CFS10] employed a spring-damper-framework, which enables the trajectories tend to converge to the intended one that was demonstrated, as shown in Fig.2.8b. However, this method brings additional challenges to maintain stability, lack of which may cause oscillations and unstable dynamics if gains were not appropriate.

**Stable Estimator of Dynamical Systems** Stable Estimator of Dynamical Systems (SEDS) is another method to represent movements but as a globally stable system that encodes the whole attractor landscape in the demonstrated data [KB10]. SEDS depends on

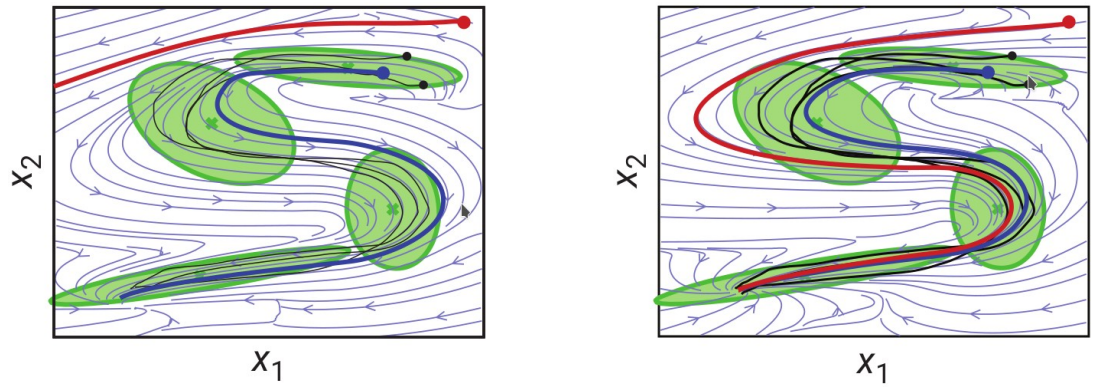



**Figure 2.8.:** The influence of a spring-damper-system on a 2 Dimensional GMR. (a) GMR without spring-damper, (b) With spring-damper system. Black lines indicate demonstraions, blue line indicates reproduction and red line indicates reproduction with different start point.(Source: [CFS10])

Gaussian mixture modules to model and generates a trajectory that is close to the demonstration. However, in contrast to the methods covered so far, SEDS focuses on global stability. A function that generates motion or trajectory is defined globally and converges asymptotically towards the target. Like GMM, SEDS comprises the mixture of Gaussian with prior, mean, and covariance as model parameters, and transition matrix is not considered [KB10]. Same GMR representation as explained earlier can be used to extract an autonomous system by computing velocity commands iteratively $\mathcal{P}(\dot{\boldsymbol{x}}|\boldsymbol{x})$ from a joint distribution of position and velocity data $\mathcal{P}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ encoded as GMM [HGC08][CL18].

$$\hat{\boldsymbol{x}} = \sum_{i=1}^{K} h_i(\boldsymbol{x})(\overbrace{\boldsymbol{\Sigma}_i^{\dot{x}x}\left(\boldsymbol{\Sigma}_i^{x}\right)^{-1}}^{\boldsymbol{A}_i}\boldsymbol{x} + \overbrace{\boldsymbol{\mu}_i^{\dot{x}} - \boldsymbol{\Sigma}_i^{\dot{x}x}\left(\boldsymbol{\Sigma}_i^{\boldsymbol{x}}\right)^{-1}\boldsymbol{\mu}_i^{\boldsymbol{x}}}^{\boldsymbol{b}_i}) \tag{2.2}$$

SEDS with GMM model and its parameters are depicted in Fig.2.9.

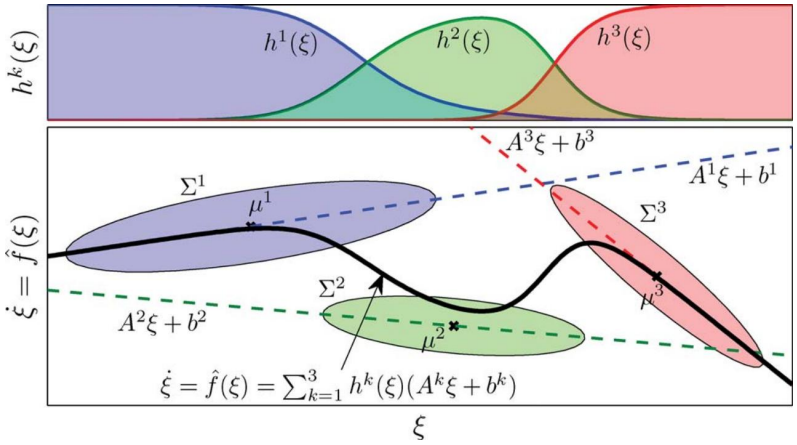



**Figure 2.9.:** Visualization of SEDS with 1D GMM and its parameters, Source:[KB11]

### Dynamic Movement Primitives

Dynamic Movement Primitives is a popular methodology for encoding movements under LfD and imitation learning paradigm in robotics. DMP is based on a dynamic system modeling approach that comprises a spring-damper system with nonlinear forcing terms, which is an approximation function[INH13]. The goal is to obtain the approximation function by learning from the demonstration data. LWR technique can be employed to obtain approximation function. DMP system comprises three elements as shown in Fig.2.10.

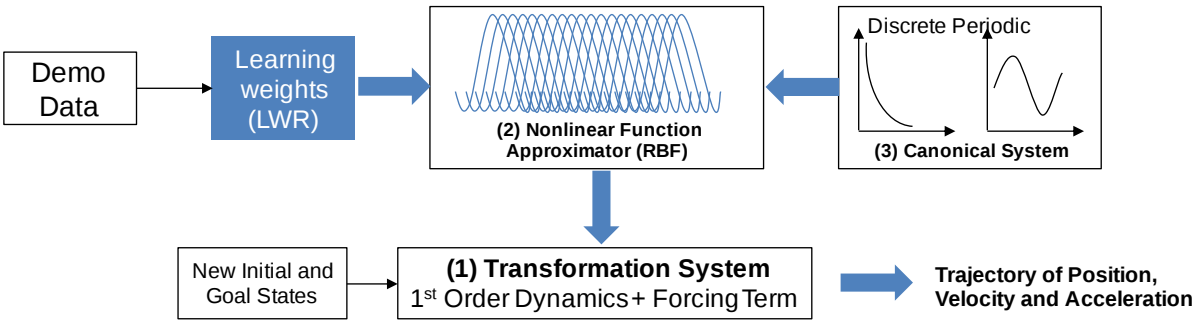



**Figure 2.10.:** Principle of Dynamic Movement Primitives, Source:[PKM13]

A stable dynamical system that is represented in the Transformation System shown in Fig.2.10 is a critically damped spring-damper model with forcing term that converts a desired force and goal position into the output commands as position, velocity, and acceleration. These commands generated at every sample timestep can be fed into an appropriate controller, as shown in Fig.2.3. A forcing term is obtained in the Transformation System from the nonlinear function approximator utilizing the phase variable generated from the canonical system. Usually, any nonlinear function approximator can be used. Most commonly, LWR with a nonlinear basis function is used. However, other approximation functions like GMM, GMR, RBF are also viable. The canonical system is responsible for substituting time with phase variable. The phase variable helps to maintain a system as time-invariant and is also used for synchronization during trajectory generation when multidimensional data is considered.

From the analysis of aforementioned methods to learn a movement primitives, as a part of this thesis, DMPs are chosen for learning the contact-based skills as DMPs are stable nonlinear dynamic systems with attractor behaviour and posses the ability to generalize to novel situations by changing the initial and goal states. Another important characteristic of DMP is that the system exhibits desired behavior irrespective of initial conditions. Its time invariance property further motivating the use of DMPs in this thesis. A detailed literature review on MPs is presented in chapter 3. A detailed explanation and working model of DMP is given in section 4.1.3

# 3. Related Work

Different methodologies exist that target skill learning for contact-based skills. Usually, these methodologies are different in their assumptions about the tasks intended to achieve or findings or focus on various learning methods. This chapter provided a survey on other methods existing in the context of LfD for skill learning, movement primitives, and control strategies focusing on contact-based tasks. Furthermore, the considered methods are classified, summarized and differentiated from the new approach taken in this thesis.

## 3.1. Learning from Demonstration

Learning from demonstration covers a wide range of topics with different learning methods and machine learning techniques. Researchers employed different viewpoints under LfD context based on their expertise, problem statement, and intended task [BG13][LRS15]. This section provides a brief study on existing research works in LfD, focusing on topics relevant to this thesis.

The term Imitation Learning was deeply studied and analyzed by Bakker et al. in [BBK96]. Bakker identified three processes involved in imitation learning as *1)Observation* to observe and perceive the demonstrated actions. Observation is similar to sensing phase, *2) Encoding and 3) Execution* as explained in Sec.2.1.

LfD can be adopted at multiple levels, like low-level motor skills that learn and reproduce movements to the high-level tasks representing symbolic reasoning. In other words, LfD can be applied from simple imitation of tasks to high-level abstractions [BG13]. To achieve robot LfD, three main challenges were identified that need to be addressed, such as *correspondence problem, generalization and robustness against perturbation.* In general, the existing research under LfD focused on solving these three challenges.

Different demonstration modalities as explained in section 2.1 should be considered appropriate methods to deal with the correspondence problem. Comparatively, Kinesthetic demonstration simplifies the challenge of correspondence problem over other methods. However, there are limitations in the Kinesthetic teaching method as the user does not execute the task but demonstrates the task by physically holding the robot. This limits to teach complex tasks as it is difficult to articulate motions [ZH18].

Another important characteristic of LfD is that robots are capable of acquiring skills that should be able to adapt to a novel situation. A common approach is to map the skills

with task variables such as initial and goal points and then perform regression to generate movements based on new task variables []. An alternative method for generalization is to encode movements considering multiple coordinate systems as explained in [UUN15] [Cal16] [PL18]. Task parameterized movement primitives as in [Cal16] encodes movements from the perspective of multiple coordinate systems. This enables the adaptation of the variations in each frame's task results in better extrapolation capabilities to adopt movements in novel situations. A similar viewpoint needs to be adopted for the contact-based tasks to consider coordinate systems to encode movements. However, in contact-based tasks, coordinate systems represents compliant frames and should be dealt with careful consideration specific to skill [MGK20] [CH18] [KGS15].

In recent years, approaches like learning by interaction or iterative learning or active learning gained considerable attention in the field of LfD [CT12]. Most of the developments in LfD concerning the development of learning and control algorithms considering enough data is available. In contrast to these methods, active learning can be used when enough data is not available to deal with novel situations. In active learning, robots update learning policies by iterative interaction with the user. The data provided can be exploited and quality can be increased iteratively by querying uncertainties while learning new skills.

The use of LfD methodologies in Human-Robot collaboration has also increased in recent years, for example, shared control behaviours with humans such as collaborative transportation of objects as presented in [EGC09] and industrial assembly tasks as given in [EGC09] [EGC09]. For further study of LfD in HRC tasks, Jangqon et al. presented a detailed comparison study and survey of LfD in Human-Robot collaboration in [Lee17].

Research Works like [ST16][TOK07] show that the use of LfD methodology for assembly tasks is one of the fastest and efficient ways to program robots from low-level tasks to abstract level planning. Takamatsu et al. in [TOK07] used LfD methodology for assembly tasks and proposed a method to identify the assembly tasks from human demonstrations. They defined the skills, sub-skills, and transitions in assembly tasks like peg-in-hole insertion and performed learning from human demonstrations. In [ZH18], Zhu et al. presented a survey and in-depth analysis of Robot Learning from Demonstrations in assembly tasks. Zhu et al. explained the use of LfD in robotic assembly and presented a detailed study on research problems like Pose estimation, force estimation, assembly sequencing. Also detailed about types of demonstration, extraction of movement primitives and metrics of imitation for robotic assembly under LfD context. This analysis on industrial assembly tasks helps us to understand the thesis requirements to reproduce contact-based skills. This thesis focus on learning contact-based skills learned from human demonstration but not collaboration and executed without iterative or interactive learning. This thesis

mainly focus on skills-based learning in contrast to generic approaches. This thesis formulate a mechanism for defining specific procedures that are more engineered for learning a specific skill than adopting a generic way.

## 3.2. Movement Primitives

Movement representation is a key research topic under Learning from Demonstration. Such representation aims to encode movements and also able to synthesize them. The term Movement Primitives is often referred to such representations as explained earlier in Sec.2.2. Movement Primitives can basically be categorized into Statistical model methods and Dynamic System model methods. Also some research works exist on the combination of these methods [PL18] [Cal16]. In this section, existing research on these methods are presented and later conclude to choose a suitable method for the objective to learn contact-based skills.

Statistical model based methods are often referred to as Probabilistic approaches that use probability distributions to encode the movements [CGB07][CFS10][RCC16][Cal16]. The variance in the distribution that defines the variability of movement obtained by learning from data is an important parameter that reflects the behavior of the motion representation while reproduction, such as, generalization and natural way of movement reproduction. Calinon et.al in [CGB07][CFS10] proposed a probabilistic approach Gaussian Mixture Regression(GMR) as explained in 2.2 for trajectory representation. Rozo et al. proposed GMR model learning using EM algorithm in [RCC16]. However, dealing with novel situations that are very far from the demonstration is problematic. Calinon et al. extended GMR with a new approach called task parameterized movement learning in [Cal16] that enables generalization capability to different situations by considering multiple coordinate systems.

Another model representation of GMR is a Hidden Markov Model (HMM) [CFS10] that encodes movement primitives in the form of states. Similar to GMM, movements are encoded with Gaussian distributions along with transition information. The states are depended upon only one parent state and follow Markov chain rule. The variants of HMM also emerged, such as incremental learning as proposed in [LO11], HMM with the local encoding of state duration for partial movements using minimal intervention control presented in [ZCC16] etc. Other variants of GMM and HMM are presented with detailed study in [CL18].

Dynamic system methods such as Stable Estimator of Dynamical Systems (SEDS) and Dynamic Movement Primitives (DMP) have gained considerable attention to represent

movement primitives. SEDS proposed in [KB11] represents movements that encode the entire attractor landscape in the state space of the data. Such representation is basically the time-invariant autonomous systems. The variants of SEDS proposed in [NS15] and [PS16] addressed the issues underlying the computation of estimating asymptotically stable dynamic system.

Dynamic Movement Primitive is trajectory-based movement representation proposed by Ijspeert et al. in [INH13]. DMPs represent a linear attraction system that is based on a spring-damper dynamic system. A monotonic time-dependent function termed as forcing term modulates the movement representation. However, instead of time as an input variable to represent trajectory, a phase variable was introduced that temporally scale the movement. The forcing term is basically represented as a weighted sum of the product of the phase variable with a basis function such as the Radial Basis Function (RBF).

DMPs are used in multiple scenarios. However, here it is focused on the use of DMPs in imitation learning as explained in [SPN05] [SPN03], where the weights of basic functions are learned using regression methods such as LWR from the demonstrated data, ideally with a single demonstration. A trajectory can be synthesized by changing the initial and goal states that enable the adaption capability of DMP to novel situations. However, the generalization capability is limited in the original formulation of DMP. An extension to the DMP proposed by Kormushev et al. in [KCC11] uses a probabilistic framework GMR for regression of forcing term in DMPs. Also, task parameterized DMP using GMM was proposed in [PL18] that adopts multiple coordinates and provides better generalization capabilities.

Although DMPs exhibit many useful properties like a generalization to novel situations such as new initial and final positions, learning by just single demonstration and temporal scaling through phase variable, additionally the sequencing of MPs and simultaneous activation of multiple MPs are important aspects needs to be incorporated for better generalization capability. However, as a part of this thesis, it is not intended to perform simultaneous activation of MPs, but here, segmentation is performed on the trajectory and MPs are sequenced one after other with continued motion. There are existing research works on the sequencing of MPs to transition smoothly from one to another as proposed in [Par17] are not considered here for simplicity. The focus is to learn contact-based skills as a primary goal. Additional improvisation, such as the smooth transition of MPs and GMR or task parameterized DMPs are considered as future work.

Though DMPs are appealing to learn position trajectories, now the question is how to encode force trajectories and combines both position and force tracking? Which is a

prime requirement of learning and reproducing contact-based skills. Kormushev et al. in [KCC11] proposed a method for combined imitation learning of position and force quantities in interactive tasks such as ironing, pressing, etc. GMMs are used to represent the movements, and variations in movements are handled by the variance parameter of GMM. At first positional profiles are learned from kinesthetic teaching and reproduced, then force profiles are learned during positional reproduction using an external sensor. The final reproduction of skill is performed with combined positional and force profiles learned earlier in two different steps. This method has a drawback that learning the profiles is performed differently. It is intended to learn the profiles simultaneously and should be combined based on skill requirements. Steinmetz et al. employed a similar procedure in [SMK15]. Position DMPs are learned over the X, Y dimensions, and Force profile using the same DMP model over the Z dimension. Conkey et al. also adopted DMPs for learning both position and force profiles simultaneously using the same model in [CH18]. This thesis work employs a similar procedure to learn positional and force quantities simultaneously from the demonstration data. Activation of respective DMPs position/force in each dimension is performed during reproduction based on specific skills.

## 3.3. Skill Learning

This section provides the study on different Skill learning methods employed in existing research, focusing on contact-based tasks to understand the necessary features to learn and reproduce the interactive tasks.

Lucia Pais et al. in [LKY13] proposed a framework for learning robot skills through motion segmentation and constraints extraction. Wherein demonstrated trajectories are segmented first and constraints such as reference frames, control variable, and motion features are extracted in order to facilitate the learning process. Scores were evaluated based on the variance in data within demonstration and between demonstrations. These scores are evaluated for segmentation, choice of reference frame and control variable to switch between position and force control. This method of segmentation and feature extraction in [LKY13] provided efficient generalization for different tasks. However, multiple demonstrations were required to extract such features. Reference frames were chosen but not extracted from demo data and segmentation was done based on the variability of data within and between multiple demonstrations. Another work from Lucia Pais et al. in [UUN15] addressed the constraint extraction for demonstrations. According to Pais et al., the task space constraints that are important to extract in order to learn and reproduce are defined as 1)Reference Frame in which to express the task variables, 2) The variable of interest at each time step like position or force, 3) A factor that modulates

the contribution of force and position in the hybrid controller. A similar approach, as explained earlier, was employed to identify task constraints.

Kober et al. in [KGS15] presented the way of learning movement primitives that depend not only on kinematic quantities but also on force interactions. Similar to the method employed in [LKY13], Kober et al. performed learning by segmentation, later computed scores to choose the Reference frame and control variable. Segmentation was done based on the Zero Velocity Crossing (ZVC) method after aligning trajectory using the Dynamic Time Wrapping (DTW). Kober et al. proposed a different way to compute scores based on the convergence behavior of the demonstration. This method also requires multiple demonstrations to evaluate scores. Each segment of the trajectory is considered as movement primitive and DMPs are considered for learning MPs. A simple hybrid position force controller proposed in [CR79] was used to reproduce learned movement primitives. Reference frames were defined as specific to the task but were not extracted from data. However, reference frames were chosen out of defined frames using computed scores.

Steinmetz et al. in [SMK15] proposed a method for simultaneous teaching of Position and Force for contact-based tasks. DMP was used to learn the trajectories as well as force profile only in the Z direction. A hybrid impedance controller along with an impedance controller was used for reproduction. Reference frames and control variables were not considered as the task was performed on fixed frames and fixed control strategy, where position control on X and Y dimensions and force control was fixed to Z direction by defining finite stiffness in X, Y, and zero stiffness in the Z-axis.

The aforementioned methods were not extracting any Reference frames from data but were chosen from the frames already defined. Conkey and Hermans in [CH18] extracted dynamic frame, which is called as Constraint Frame (CF) by learning force profiles. Trajectory segmentation is not done here. However, a selection matrix is defined to choose a control strategy in constraint frames. The hybrid position-force controller is adapted with dynamic constraint frames incorporated in it. Also, Conkey et al. contributed a new methodology for extracting constraint frames and extended DMPs to facilitate contact tasks where contact points can change dynamically.

Another work on learning force-relevant skills from the human demonstration is presented by Gao et al. in [GLX19]. A mathematical representation of Force-relevant skills is proposed as a function of position, velocity, interaction forces, and task constraints. Skill acquisition was meant to find the internal relations of these variables from multiple demonstrated data. The encoding of demonstrated data was done using GMM and GMR is used for the prediction. Task execution was performed on an Adaptive hybrid force position controller based on admittance control.

Manschitz et al. provided a method for Learning sequential skills under force interactions in [MGK20]. A dynamical system with a linear attractor was chosen as MP and a hybrid position force controller similar to Kober work in [KGS15] was used. A novel task segmentation method was proposed called DND. At first, the trajectory is segmented into multiple segments using Zero velocity crossing, but segments are grouped again using the DND algorithm to form movement primitives. Sequential skills were constructed from segments from DND and switching between skills was performed by MP sequence learning. Experiments were done on tasks like box flipping, box stacking and unscrewing the bulb. The algorithm proposed in the paper chooses the reference frame and the control in each MP. However, S.Manschitz et al. focused on the segmentation of the demonstrated task to obtain MPs and sequencing the MPs to achieve learning an arbitrary skill. This thesis work aims to develop a framework to learns the skills which are predefined and in contrast to a methodology employed in [MGK20] is to learn any arbitrary skill, it is planned to employ a similar procedure to learn a skill but in a way that parameters in each step of the procedure is more engineered specific to the predefined skill.

## 3.4. Hybrid Position-Force controllers

The main goal of the project is to learn force-based or contact-based skills. A controller like impedance control is not sufficient to implement learning for compliant tasks. A simple hybrid position-force controller that offers a selection of position and force control in each dimension, initially proposed by Craig et al. in [CR79] is one of the suitable methodologies that can be employed for contact-based tasks. As proposed in [LKY13] [UUN15] force controller is embedded along with the position controller to achieve learning in contact-based LfD context.

A similar control strategy was employed in [KGS15] [MGK20] based on task level inverse dynamics approach initially proposed in [LGM12] with a hybrid control strategy. Steinmetz et al. in [SMK15] also adopted a similar control strategy to teach and reproduce in-contact tasks. Here, a hybrid controller with impedance controller as position control and PI controller for force control was adopted. Steinmetz et al. extended the hybrid control strategy with an additional velocity controller to track the contact point. However, controller switching was fixed to Z-axis. In contrast, the requirements are to have more degree of freedom to choose the controller switching in other dimensions also.

Gao et al. in [GLX19] also adopted a similar control strategy, an Adaptive Hybrid Force position control, to learn force-relevant skills from human demonstrations. Gao et al. extend the simple PI Force controller with additional admittance control that generates a velocity command.

Hybrid control strategies aforementioned under LfD and contact-based tasks are suitable when tracking interaction forces in simple Cartesian coordinates. However, these strategies are not sufficient when a task frame undergoes a transformation. Conkey et al. in [CH18] addressed a similar issue based on [Kha87], wherein dynamically changing constraint frame was embedded in control equation to transform Selection matrix defined intuitively in constraint frame.

Marin et al. in [MW16] proposed a unified hybrid position force controller based on the Kinesthetic filtering method proposed in [LD88] and extended the approach for non-invariance filtering in [ABM90]. In [MW16], addressed that when a compliant frame undergoes a rigid transformation, the commands issued in the compliant frame need to be transformed, in other words, kinesthetically filtered.

The aim is to develop a generic hybrid controller that can be used for most of the skills. To do so, some of the prime requirements are defining control parameters like stiffness parameters and selection matrix in the compliant frame and such rigid transformations of the compliant frame should be considered in the control loop. It is required to adopt a control strategy similar to the strategy employed in [CH18] [MW16] that allows the transformation of compliant frames. However, in [CH18], only rotational transformations were considered. The goal is to adapt the rigid transformations, including translations. In contrast to [CH18], It is intended to use an impedance controller for position control and a PI controller for force control in this thesis work.

Marin et al. in [MW16] adopted the rigid transformation and an Impedance controller was employed. However, In contrast to the requirement to define the stiffness matrix in CF, Marin et al. have not considered the transformation of the stiffness matrix as it was fixed to the base frame. In the subsequent chapters, a strategy for a Hybrid position-force controller capable of rigid transformation and the transformation of control parameters is derived.

## 3.5. Summary

In summary, to learn and reproduce contact-based skills, it is considered predefined skills that are more common in industries that involve interactive tasks. This thesis employs an LfD methodology that is similar to the methods surveyed in assembly tasks [ZH18]. Since DMPs exhibit many useful properties like generalization, learning by single demonstration, and temporal scaling, as well as proven methods to use DMP for learning force profiles. DMP as movement primitives is considered and performs learning of position, orientation, and force profiles simultaneously. The adaption and implementation procedures are explained in detail in Chapter 4. Later, from the analysis of Skill Learning methods presented earlier in this chapter, features are identified that represent skills and extract for each predefined skill. From the viewpoint of controller implementation methods discussed earlier in this chapter. A hybrid position force controller and further extend to the control strategy that is capable of adapting a rigid transformation of compliant frames.

# 4. Approach

In order to develop a skill-based learning framework for LfD, the methodology considered for this thesis is based on a segmentation and encoding process. The skill which is demonstrated can comprise of multiple MPs and therefore can be represented by sequencing them. Here, the main idea of learning a skill is by performing trajectory segmentation and encode each segment to a movement primitive specific to the skill. Each segment can be represented as one MP. Different types of MPs are already discussed in section2.2. As shown in [CL18], DMP, GMR with GMM and HMM, etc., are some of the MP representations that are more suitable for contact-based tasks. To extract these MPs, trajectory segmentation is a significant step. In this chapter, the segmentation methods employed to segment demo trajectories into a meaningful set of MPs are addressed. Detailed framework and comparison of different segmentation methods are provided in [LKK16].

After performing segmentation and extracting points for the encoding of MPs, compliance frames are needed to be extracted in order to apply forces intuitively to interact with the environment. By fitting geometrical constraints specific to skills, compliant frames can be extracted based on local coordinates of the constraint for each segment in each predefined skill. Analysis and implementation for the best fit of constraints specific to the skill will be addressed in this chapter.

Suitable parameters and features specific to each skill are identified and extracted for reproduction, which should be robust to noise and environment dynamics. Parameters like Normal forces and Stiffness parameters for impedance control are computed at this step. As a part of this thesis, the identification and extraction of such parameters are performed.

A control strategy will be derived for a hybrid controller to track and control position and force quantities. The controller employed is capable of reproducing interactive skills considering compliant frames, parameters, and features specific to each skill.

The approach to develop a skill-based learning framework for contact-based tasks is employed in two phases :

1. Skill Independent Approach: Employ generic methods of learning without considering skill-specific requirements.

2. Skill-based parameterization: Establish approaches that are specifically framed for skills. Also, identify and extract skill-specific parameters for the modules implemented in Phase I.

## 4.1. Phase I: Skill Independent Approach

At the initial phase, a theoretical analysis is performed and implementation of the required modules is provided, which are independent of skills. The workflow overview of Phase I is shown in Fig.4.1

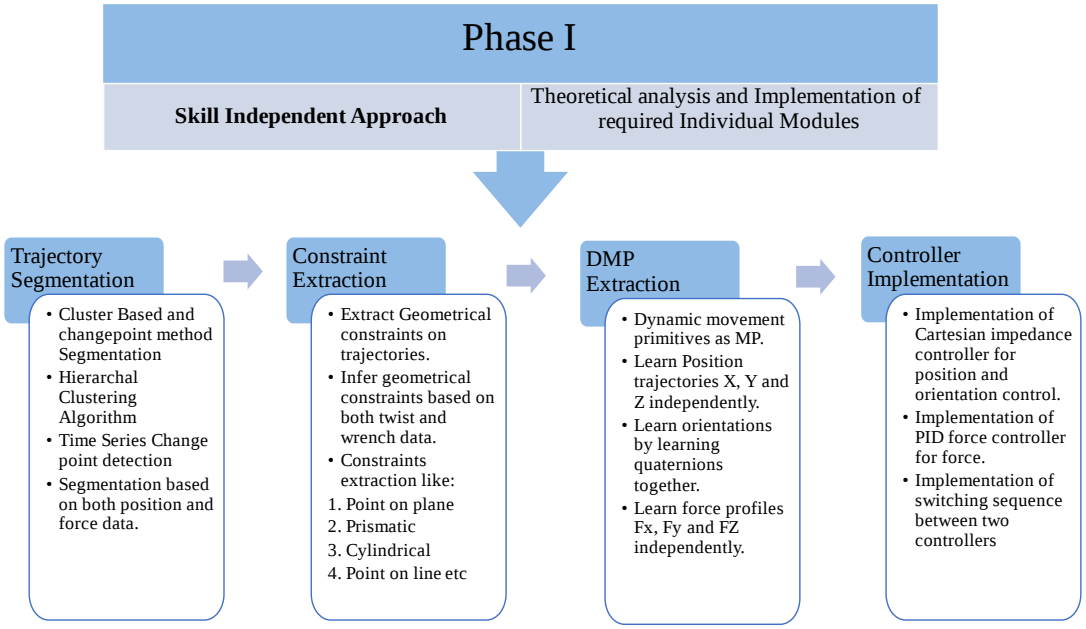



**Figure 4.1.:** Phase I

### 4.1.1. Trajectory Segmentation

Trajectory segmentation is an important step to learn a set of movement primitives of a complex task. Often a single MP is considered for learning an entire trajectory. However, by sequencing or blending a set of MPs, the complex tasks can be learned with more accuracy. The segmentation procedure in order to obtain MPs is often called Motion Segmentation and is explained in detail along with its significance in imitation learning in [LKK16]. Lin et al. in [LKK16] also addressed different online and offline motion segmentation methods and algorithms and framed novel methods considering factors to be incorporated for motion segmentation. However, the methods explained in [LKK16] are not limited to the LfD context.

Existing research on trajectory segmentation proposed in the LfD context includes different types of segmentation methods 1) Classification using prior data by Manual Segmentation 2) Zero Velocity Crossing(ZVC) approach 3) Clustering using unsupervised learning, and 4) Change Point Detection based segmentation.

**Manual segmentation** requires expert knowledge to segment the trajectories meaningfully. There are multiple methods proposed in manual segmentation by sensors attached to the robot, using visualization tools. Multiple machine learning algorithms are also proposed for segmentation by using prior labels performed by manual segmentation [LKK16]. However, for a huge amount of data and every new skill, this process of labeling using manual segmentation to train machine learning models is cumbersome. This approach is not used in this thesis to facilitate user-friendly robot teaching.

**Zero Velocity Crossing (ZVC)** approach for segmentation as used in [KGS15] [MGK16] [MGK20] is a commonly used procedure as it is easy to implement and is a more robust way of segmentation. The zero-velocity crossings are identified from the demonstrated data. Segmentation points are considered where velocity crosses zero. Recent advances using the ZVC approach like in [MGK20] using DND to assign segments to MPs outperforms other approaches. However, while teaching a robot, the user needs to consider when to stop the robot to differentiate between motion segments. This demonstration method requires expert knowledge about the task or skill that is being demonstrated.

**Cluster-based Segmentation** methods are considered to be an automatic way of segmentation. Here, unsupervised learning methods to segment trajectory points by grouping similar points [KTN08] is used. For force-based tasks, considering force to influence the segmentation can be facilitated in clustering methods. In [XLE18], Xie et al. proposed segmentation by spatial and temporal way of hierarchical clustering considering both position and force data from the demonstration. This method is well suited for this thesis goals, which facilitates a user-friendly approach. The user need not consider additional parameters during the demonstration. However, careful skill-specific parameterization and tuning are necessary before developing a skill library.

**Change Point Based Segmentation** approach in general deals with segmenting univariate and multivariate time series data [LT06]. As the demonstrated trajectory data can be considered as a multivariate time series, change point detection is used to address the motion segmentation to obtain MPs [NOA] [NOA15]. In the LfD context, Niekum et al. in [NOA15] addressed online change point detection based segmentation method for demonstration data using Bayesian Information Criteria for maximum likelihood approximation.

In reference with [LKK16], comparing different methods and considering the factors for contact-based skill learning such as automatic segmentation, robust segmentation of patterns, and meaningful representation of motion, Cluster-based and Change point based segmentation methods are considered here as more suitable methods for learning compliant skills.

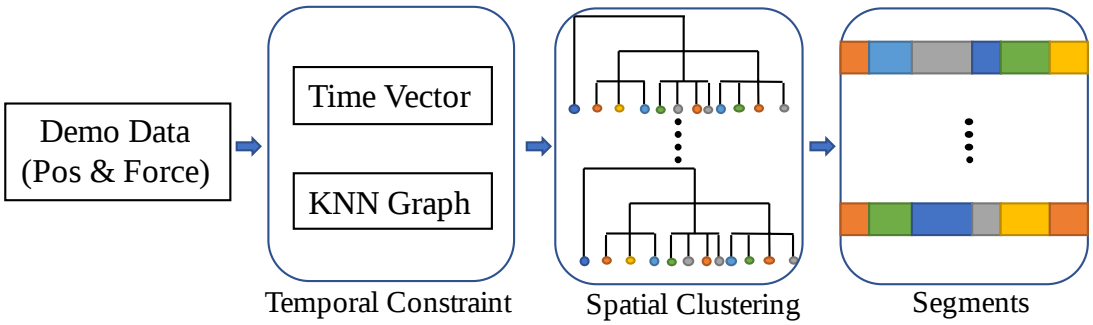



**Figure 4.2.:** Unsupervised Hierarchical Clustering based segmentation

### Implementation of Cluster Based Segmentation

Clustering is a grouping mechanism of similar data, which is an interesting feature to group multiple patterns. As explained in [XLE18], Agglomerative Hierarchical clustering is one such unsupervised clustering algorithm that is capable of grouping similar features in demonstration data without defining a required number of clusters beforehand. Automatic grouping is done based on predefined metrics called linkage such as ward, single, complete, and average metrics. Each group obtained from demo data can potentially be represented as an MP. However, it is not straightforward to represent each group as one movement primitive because it is important to understand that grouping trajectory data is not the same as segmenting trajectory. It is because non-neighbor samples are also grouped, where segmentation of trajectory is intended to group similar patterns among neighboring samples of data. Hierarchical clustering usually groups similar data only in the spatial domain, which results in grouping non-neighbor samples. Therefore, segmenting trajectory data, which is a time series data, should consider temporal constraints to group only neighboring data into segments.

The issue of handling spatial and temporal clustering is addressed in [ZMT13] [XSX16] [XLE18] for segmenting trajectory data in the robotics domain. A similar methodology is adopted in this thesis to segment demo data using Agglomerative hierarchical clustering in the spatial domain but for temporal constraint, in contrast with methods used in [ZMT13] [XSX16] [XLE18], here two different methodologies are considered for simplicity. They are 1) Time Vector as an additional feature, and 2) KNN Neighbors as a prior, as shown in Fig.4.2.

**Time Vector** as a temporal constraint is a simpler way to deal with the segmentation of trajectories. In this approach, an additional time vector, a column with an increasing sample index number of rows, is added to the demo data with position and force quantities. The modified data is now passed to the Agglomerative clustering algorithm as shown in

Alg.1. The output of the algorithm is a set of trajectory segments considering both position and force data which can be represented as a Movement Primitive.

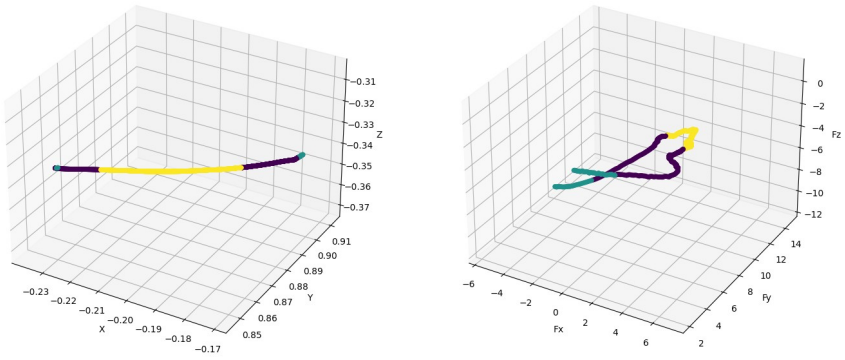
**KNN Neighbour** as a prior for connectivity constraint which defines the neighbouring samples. The KNN neighbors as a connectivity matrix are passed along with the data to the Agglomerative clustering algorithm and therefore, only neighboring samples of data are grouped together. The output of the algorithm gives a set of segments of demonstrated trajectory data.

Figures 4.3a 4.3b 4.3c shows the comparison of clustering of trajectory data with and without time constraints. This figure gives a detailed understanding of the requirements of temporal constraint in trajectory segmentation as also addressed in [ZMT13] [XSX16] [XLE18].

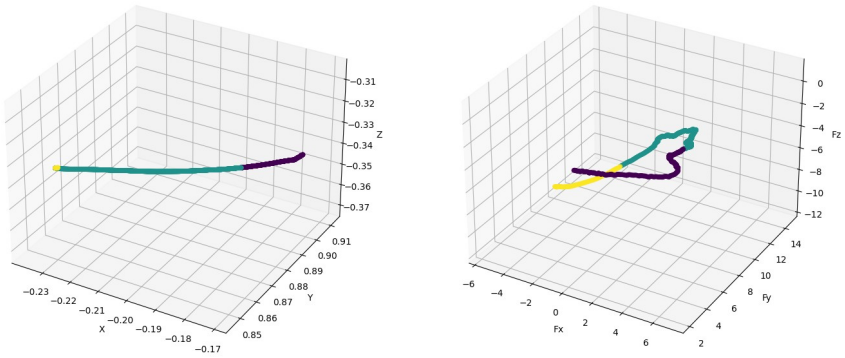
**Implementation of Change Point Based Segmentation**

Niekum et al. in [NOA15] introduced an algorithm for online change point detection also convincingly applied on LfD data for robots. The algorithm performs Bayesian Online Changepoint detection by estimating the maximum likelihood parameters for each segment. The algorithm shown in Algorithm:2 from [NOA15] called CHAMP (Change Point Approximate Model Parameters) is directly adapted in this thesis without further modifications. The paper [NOA15] also demonstrated experimentally to infer motion information from demonstration trajectory data under LfD context. Figure 4.3d shows the results of change point detection based segmentation of demo trajectory.

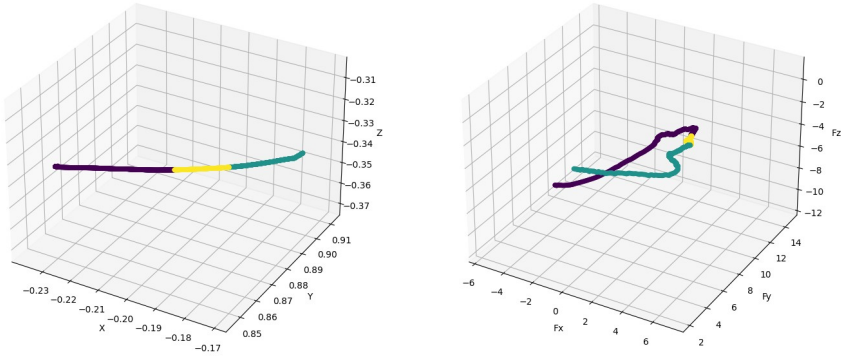
A suitable method for segmentation is chosen based upon the task requirements, the complexity of trajectories and skills. For instance, for a sliding skill, cluster based segmentation is more suitable to segment trajectories based on position and force data. On the other hand, in a press skill that exhibits more discontinuity in the data based on forces, the change point method is suitable for segmentation. The choice of segmentation method may also vary within the skill-based upon multiple parameters such as demonstration method, environment, etc.



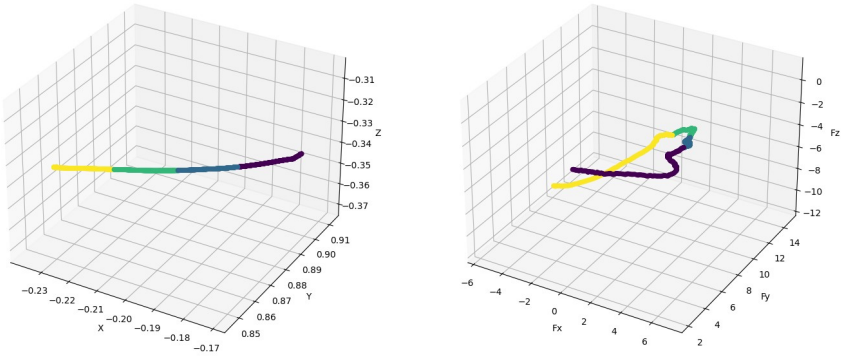

**(a)** Hierarchical Cluster based segmentation without Time Constraint




**(b)** Hierarchical Cluster based segmentation With KNN Constraint




**(c)** Hierarchical Cluster based segmentation with Time Vector




**(d)** Change Point based Segmentation

**Figure 4.3.:** Comparison of different segmentation methods

### 4.1.2. Constraint Extraction

Contact-based skills, also known as compliant skills, are skills that have controlled motion and maintain the contact forces. To learn such complex skills, demonstrated tasks are represented in feature-based representation and specific constraints need to be extracted depending on tasks. As stated earlier, learning a compliant skill is learning the movements along with the force applications. As such, constraints such as constraint frames needs to be extracted in which forces can be applied intuitively during reproduction.

Researchers have done extensive work in extracting constraint frames where force application is involved. Such constraint frames are also called compliant frames(CF). Ureche et al. in [UUN15] extracted force information, reference frames and constraints from demonstrated data to learn compliant skills. However, the constraint extraction is limited to the reference frame, task variable, and stiffness modulation. The reference frames are chosen from the predefined frames. A similar approach is employed in extracting constraints framed in [LKY13] which are defined manually with prior knowledge for every task. The aim is to automatically extract such reference frames for predefined skills without prior knowledge of the user.

The extraction of the frame needs to be pre-programmed for specific Skills beforehand while developing skill templates. Conkey et al. in [CH18] extracted such constraint frames dynamically by learning the contact force trajectories obtained during the demonstration. Although the constraint frames are extracted dynamically by learning force profiles, the reference frames extraction is way too generic to implement different skills. For example, this method of dynamic extraction can be applied to contouring skill, but sliding skill doesn't require dynamic extraction of constraint frames. In sliding skill, constraint frame can be fixed to a plane geometry on which sliding is being performed. The constraint extraction methods should vary with respect to every skill and should be designed and pre-programmed beforehand while constructing abilities. As stated in the goals, the end-user without any prior knowledge shouldn't be given a choice to choose or construct frames, which has to be done beforehand during skill construction.

It is required to have a more specific way to extract constraints to define compliant frames for learning force applications. Such a specific way of constraint extraction is identified in [SZG18]. Guru et.al in [SZG18] presented an approach for inferring geometric constraints in human demonstrations to represent kinematic constraints of motions. The proposed approach first fits constraint models based on only kinematic quantities and evaluates them individually using position, force, and moment criteria. The work is limited to only extracting such constraints on human demonstration but not implemented in learning scenarios. In this thesis, the constraint extraction based on methods proposed

in [SZG18] is adapted directly and extended to learn the constraints for predefined skills and subsequently used for learning compliant frames. From the paper, the important geometrical constraints identified from human demonstrations are as follows:

**Fixed Point Constraint:** which represents a single point that is constrained to an environment. For example, a ball joint can be considered as a point constraint where there are no linear motions but only orientations.

**Point on Plane Constraint:** is a constraint of a motion to a plane without only two linear motions on a plane are considered and other DOF of motion are ignored.

**Concentric cylinder constraint:** is motion constraint about an axis, where transnational motion is permitted along an axis and orientation is allowed about the axis. For example, peg in hole insertion task. For learning skills for peg-in-hole insertion, this geometrical constraint can be suitable to learn compliant frames.

**Planar Constraint:** similar to Point on plane constraint but allows the orientation on the plane along with motion on a plane. For example, a sliding skill performed by the robot should apply a constant force on a plane surface. Planar Constraint is suitable to extract a compliant frame to perform a sliding task.

**Prismatic Constraint:** is the representation of only transnational motion without rotational movement. For instance, tasks like pulling a drawer.

**Axial rotation Constraint:** restricts all translational motion but permits only one rotation about any point. Extracting this constraint extracts the hinge axis about which the rotational motion is performed.

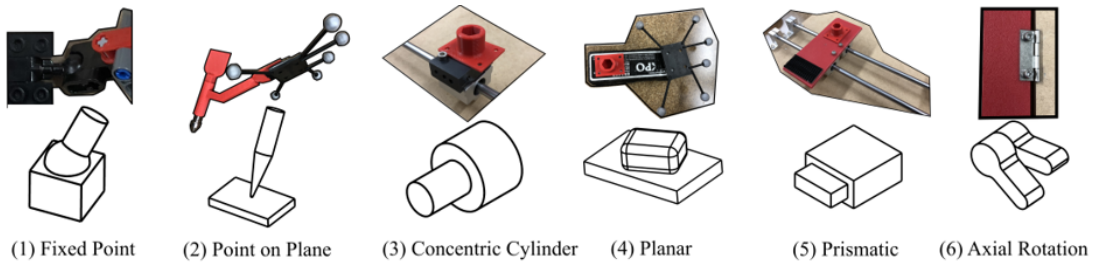



**Figure 4.4.:** Illustration of different constraints, Source:[SZG18]

### Extraction Procedure of Compliant frame for skills

Constraints are identified for specific skills based on the force application. The constraint frames specific to contact-based tasks are explained in detail for specific tasks in [BD96] as "Task Frame Formalism" or "Compliance Frame formalism." [BD96] gives us basic theoretical understanding and acts as a basis for defining CF for specific tasks. Further

analysis and identification of constraints for specific skills are explained in detail in the next chapter under feature identification and extraction.

Identified constraints for skill are extracted from the demo data of a skill. For instance, planar constraints extracted for slide skill using methods provided in [SZG18] are used to extract compliance frames as shown in the figure 4.5.

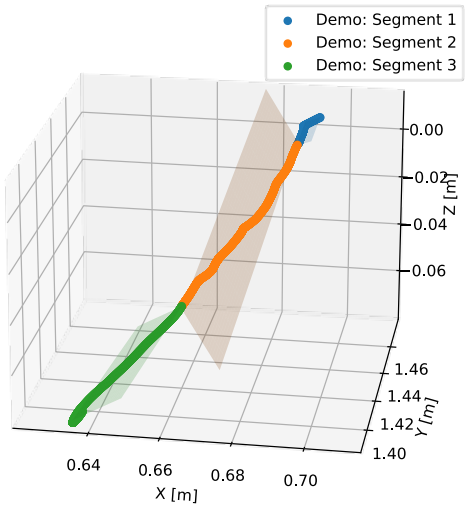



**Figure 4.5.:** Illustration of Point on Plane constraints extracted for Slide skill

### 4.1.3. Dynamic Movement Primitives Extraction

As explained in section 2.2, Movement Primitives encode motion in a mathematical model, which forms a basic building block, when arranged in sets of sequences, represent a complex motion task. Movement primitives in the context of learning skills are extensively used in recent years for learning tasks or skills. For further references, a detailed literature review of Movement primitives is explained in Chapter:3. Dynamical Movement Primitive (DMP) in [INH13] [SPN05] [SPN03] is one of the well known movement primitives in LfD context and is considered in thesis.

DMPs are modeled based on nonlinear dynamic attractor systems, where motor skills are learned from demonstrated data to model MP and can be utilized to reproduce learned motion. DMPs can approximate the trajectories with changes in initial and goal states during reproduction. However, the accuracy of approximation highly depends upon the method used to learn the function approximator and complexity of the trajectory. Advantages and implementation of Learning DMPs for imitation learning are very well portrayed in [SPN05]. In contact-based applications, DMPs are used in multiple scenarios for learning movement primitives. For example, Steinmetz et al. in [SMK15] utilized DMP

for learning position, orientation, and force profiles for in-contact tasks. Conkey et al. in [CH18] also used DMP for learning the position, force, and also dynamic constraint frames.

**DMP Model:**

The DMP model comprises of three basic elements: *1)Transformation system 2)Function Approximator and 3)Canonical System,* [INH13] [SPN05] [SPN03] [SMK15]. For this thesis, the following formulation is adapted similar to [CH18].

*1) Transformation System* represents the first order critically damped dynamical model as a spring-damper system shown below:

$$\tau \dot{\boldsymbol{v}} = \alpha(\boldsymbol{g} - \boldsymbol{x}) - \beta \boldsymbol{v} - \alpha\left(\boldsymbol{g} - \boldsymbol{x_0}\right)\boldsymbol{s} + \alpha f(\boldsymbol{s}) \tag{4.1}$$

$$\tau \dot{\boldsymbol{x}} = \boldsymbol{v} \tag{4.2}$$

System is set to critically damped with appropriate choice of $\alpha$ and $\beta$ usually, $\beta = \alpha/4$. $\boldsymbol{x}$ is a state variable which is being traced with initial value $\boldsymbol{x_0}$ and goal value $\boldsymbol{g}$.

*2) Function Approximator:* is a non linear function which is learned from the data to approximate the trajectory as a set of normalized linear combinations of basis functions which is shown below:

$$f(\boldsymbol{s}) = \frac{\sum_i w_i \Psi_i(\boldsymbol{s})}{\sum_i \Psi_i(\boldsymbol{s})} s \tag{4.3}$$

$$\Psi_i(\boldsymbol{s}) = \exp\left(-h_i\left(\boldsymbol{s} - c_i\right)^2\right) \tag{4.4}$$

$f(\boldsymbol{s})$ a function approximator with linearly weighted weights $w_i$ onbasis function $\Psi_i(s)$. The main objective of learning DMP is nothing but learning $w_i$. In this work these weights are learned using Linear Weighted Regression (LWR). Other approximator methods like GMR are also possible as used in [Cal16][PL18]. Basic radial basis function is chosen for $\Psi_i(s)$ which can approximate trajectory segments in most of the cases. For complex trajectories, segmentation methods as explained in previous section reduces the complexity to learn the DMP.

*3) Canonical System:* ensures the synchronization of state space variables as the DMP isolates each dimension of the state variable. Synchronization is carried out by replacing time variable with canonical variable. The equation of the canonical system is represented as follows:

$$\tau \dot{\boldsymbol{s}} = -\alpha_s \boldsymbol{s} \tag{4.5}$$

**Cartesian DMP model for Orientations:** The above DMP system with LWR approximator is well placed with Cartesian coordinates as each cartesian coordinate can be represented as one state variable and can be isolated or decoupled using phase variable in canonical system without any further modifications. However, a special treatment is required for handling orientation quantities when representing in unit quaternions. Because, a quaternion is a vector of four quantities which represents orientation in 3D space, but each quantity cannot be decoupled like Cartesian coordinates. In order to handle this issue, a generic form of DMP is introduced called CDMP especially to deal with orientation and torque quantities in [UNP14] [KGN17]. This form of DMP is also used in [CH18] for learning orientations of trajectory. The CDMP for orientations can be represented as follows:

$$\tau\dot{\boldsymbol{\omega}} = \alpha_\omega \delta\left(\mathbf{q}, \mathbf{q}_d\right) - \beta_\omega \boldsymbol{\omega} - \alpha_\omega \delta\left(\mathbf{q}_0, \mathbf{q}_d\right) \boldsymbol{s} + \alpha_\omega \mathbf{f}(\boldsymbol{s}) \tag{4.6}$$

$$\tau\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\omega} * \mathbf{q} \tag{4.7}$$

In the above representation, equation of transformation system in equation 4.6 is simillar to equation 4.1 represented in DMP. But without phase variable. All the quantities are learned together without isolation. $\delta\left(\mathbf{q}, \mathbf{q}_d\right)$ is a quaternion error or quaternion difference function defined as $\delta\left(\boldsymbol{q_1}, \boldsymbol{q_2}\right) = 2\log\left(\boldsymbol{q_2} * \bar{\boldsymbol{q}}_{\mathbf{1}}\right)$ [CH18] [UNP14].

The adaptive capability of DMP for change in initial and goal points by approximating the trajectory is well suited for noncontact-based skill learning. Nevertheless, in order to learn contact-based skills, additional modifications are required to adapt dynamics in the environment for interaction tasks. The feedback from the robot is considered to track the contact state at each time step parallel to the task execution. The initial and goal points are adapted based on the contact state.

**Implementation of DMP Extraction:**

In Cartesian spaces, the DMP model isolates each dimension of the states space. For example, in a specific skill X, Y, Z, Fx, Fy, Fz are position and force profiles respectively needed to learn, such that as a sum, six different DMPs for each variable will be modeled and the synchronization is carried out using canonical phase variable. On the other hand, for learning orientation quantities, a combined DMP system is learned and LWR is used as a regressor to learn function approximator. The overview of the DMP module implementation for the thesis work is shown in Fig.4.6

Position and force profiles required for skills are learned using normal DMP and orientations are learned using Quaternion DMP. For skills, where contact requirements such as dynamical initial and goals are needed, dynamically adapted the states with contact state

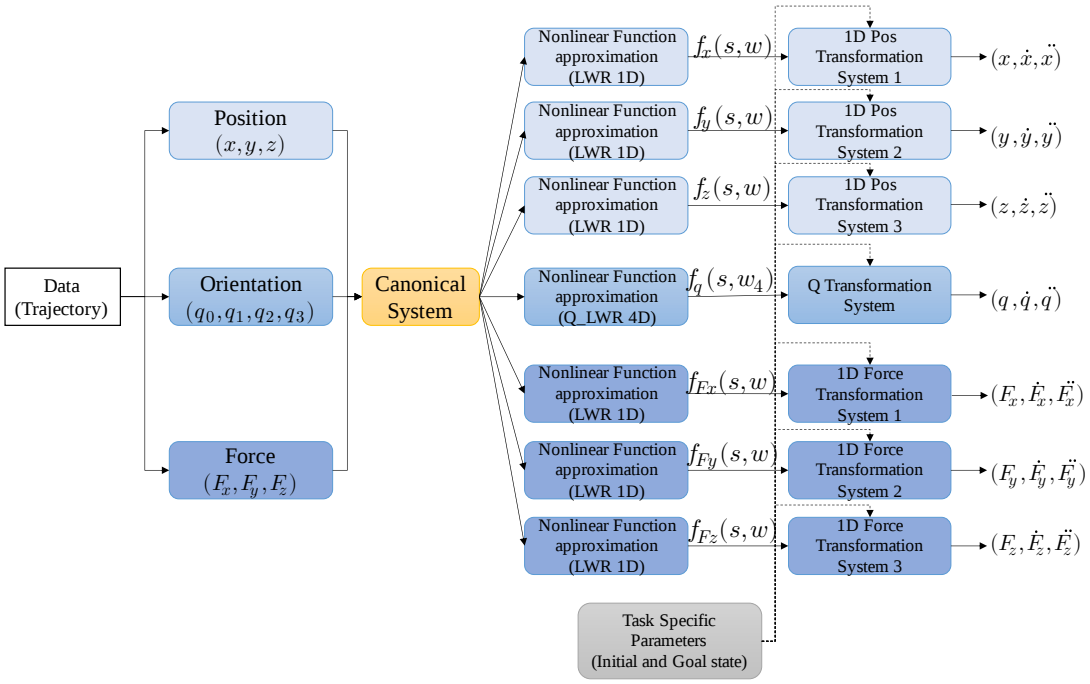
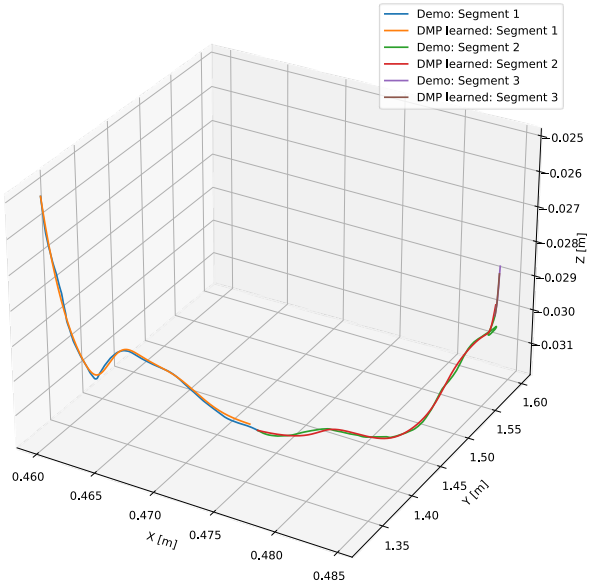
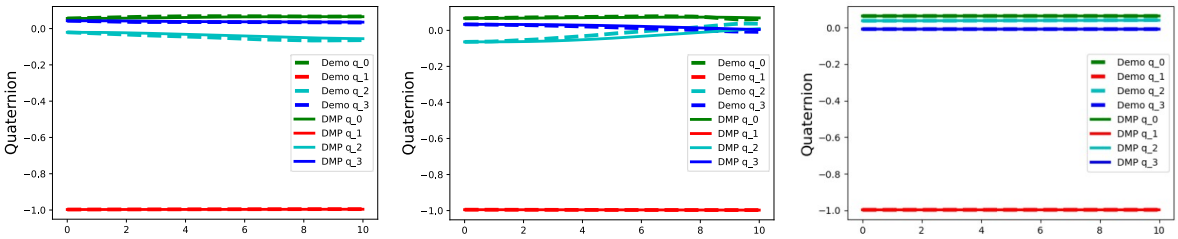



**Figure 4.6.:** DMP Extraction for Position, Orientation and Force Quantities




**(a)** DMP extraction of Position of a trajectory with three segments




**(b)** DMP extraction of Orientation of a trajectory with three segments

**Figure 4.7.:** Example for DMP Extraction for Position and Orientation

feedback from the robot. As far as skills predefined until now, there is no requirement of learning torques. Figure 4.7 shows an example for DMP extraction for position and orientation of a sample trajectory with three segments.

### 4.1.4. Hybrid Position-Force Controller

In order to learn and reproduce contact-based skills, tracking kinematic quantities are not enough but also interaction forces and moments need to be tracked and controlled. For controlling a robot, multiple control paradigms have emerged in order to track position, velocity, and joint torques. Robust robot controllers like Impedance controller, Inverse dynamic controller are able to control robots by tracking Joint torques of the robot. However, such controllers are not able to handle interaction forces with the environment and hence not suitable for performing compliant tasks. A detailed study on comparison of different contact-based and contact-free controllers is explained in [XHL19] which gives a detailed understanding of compliant control strategies such as High-Level Compliant control and low-level robot motor control in the LfD context. In order to control both position and interaction forces, a high-level hybrid position-force controller similar to the controller implemented in [CH18] [KGS15] [MGK20] needs to be employed. In order to implement such a hybrid controller, for tracking position and orientation, an Impedance controller is chosen and for tracking force and torques, a simple PI Force controller is chosen.

**Impedance Controller:**

Impedance controller is first proposed by Hogan et.al in [Hog84] which is modeled based on non-linear virtual spring damper system which regulates the acceleration, velocity, position, and forces, as a result it regulates the mechanical behavior of a robot. In impedance control main objective is to achieve the desired impedance of the robot movement. The control law can be written as follows:

$$\boldsymbol{\tau_{ic}} = J^T(\boldsymbol{q}) \left( \boldsymbol{K_c}(\boldsymbol{e_p}) + \boldsymbol{D_x}\boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{\theta}} + g(\boldsymbol{q}) \right) \tag{4.8}$$

$$\boldsymbol{e_p} = \boldsymbol{x_d} - \boldsymbol{x_{msr}} \; ; \boldsymbol{x} = \begin{bmatrix} position \\ orientation \end{bmatrix}_{6x1} \tag{4.9}$$

Here $\boldsymbol{\tau_{ic}}$ denotes Joint torques as commands to robot from Impedance Controller, $\boldsymbol{J(q)}$ denotes Jacobian of the robot, $\boldsymbol{K_c}$ denotes stiffness, $\boldsymbol{e_p}$ is the error of desired and measured vector of *position* in Cartesian coordinates and *orientation* in Euler angles as $\boldsymbol{x_d}$ and $\boldsymbol{x_{msr}}$ respectively. $\boldsymbol{D_x}$ and $g(\boldsymbol{q})$ denotes Damping and gravity compensation terms of nonlinear dynamical system. The basic control loop of the Impedance Controller can be depicted as shown in the figure 4.8

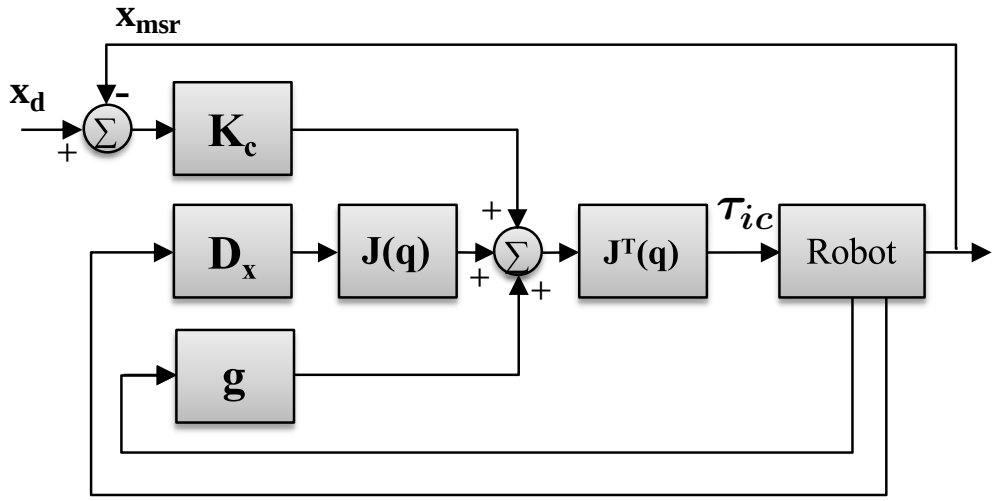



**Figure 4.8.:** Impedance Controller

In the control loop shown in Fig.4.8, the minimal version of the Impedance controller is depicted with position and orientations in cartesian coordinates and Euler angles, respectively. Handling Euler angles is problematic and highly unstable to regulate the forces and moments in stiffness module $\boldsymbol{Kc(e_p)}$. Now the question of representing orientations of the end effector is a challenging problem that needs to be addressed in order to design a robust controller. Caccavale et al. in [CNS99] presented a detailed comparison of different representations for orientations such as Euler angles, Rotation matrix, Axis angle representation, etc. The paper [CNS99] explains evidently that the instability is caused due to Euler angle representation.

Also, another important extension of the Impedance controller called Spatial Impedance controller proposed in [FB97] which is based on spatial stiffness, solved the issue of robust representation of orientations. Stramigioli and Duindam made further extensions in [DSD01] to obtain force and moments of stiffness module by modeling it as Variable Spatial Springs. In a simple impedance controller, forces and torques together are called wrenches, are obtained by simple multiplication of Stiffness matrix $\boldsymbol{K_c}$ with the error of pos vectors $\boldsymbol{x}$. In contrast to simple impedance controller, in [DSD01], forces and moments are obtained with complex mathematical calculations by modeling variable spatial springs for the stiffness module. This method is proven to be one of the stable configurations that can be included in the impedance controller. The Variable Spatial Spring-based Impedance controller, which is already developed, is adopted directly in this thesis for tracking of position and orientations of the robot end-effector. The overview of the Variable spatial Spring-based Impedance controller is shown in the figure 4.9. The implementation of the controller is done entirely in the MATLAB Simulink package, and the interface of the package with other modules is explained in detail in section 5.1.

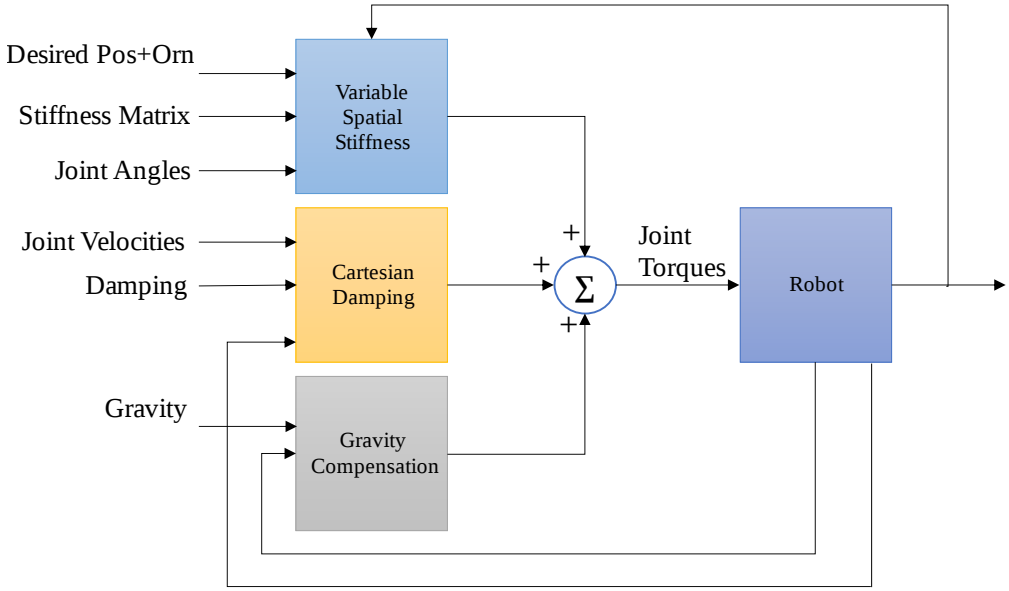



**Figure 4.9.:** Overview of Variable Spatial Stiffness Impedance Controller

**Force Controller:**

As stated earlier, in order to reproduce contact based skills, compliant tasks are executed to interact with environment. Hence, along with impedance controller a force controller is needed to perform compliant tasks. A simple PI control law as written below is adapted for regulating of interaction forces with environment.

$$\boldsymbol{\tau_{fc}} = \boldsymbol{J^T(q)} \left( \boldsymbol{K_p e_f} + \boldsymbol{K_i} \int \boldsymbol{e_f} dt \right) \tag{4.10}$$

$$\boldsymbol{e_f} = \boldsymbol{f_d} - \boldsymbol{f_{msr}} \; ; \boldsymbol{f} = \begin{bmatrix} force \\ torque \end{bmatrix}_{1x6} \tag{4.11}$$

Here $\boldsymbol{\tau_{fc}}$ denotes Joint torques as command to robot from Force controller, $\boldsymbol{e_f}$ denotes error of desired and measured vectors of $forces$ and $torques$. $\boldsymbol{K_p}$ and $\boldsymbol{K_i}$ are proportional and integral constants of PI controller respectively. The overview of control law is shown in Fig.4.10.

However, above mentioned control strategy for force control is a direct control method which exhibits highly unstable behaviour while interacting with environment. It is hard to tune the parameters to maintain stable contact and apply stable contact forces on to the environment. Also, with change in environment, even with well tuned parameters, controller exhibits unstable and oscillating behaviour. Similar issues is address and compared with different force control algorithms by Wilfinger in [WL18]. Wilfinger extended PID controller for force control with Integral Error scaling and Force signal clipping to limit the windup of integrator. These modification exhibited significant improvement to maintain a stable contact and apply stable forces without oscillations on to the environ-

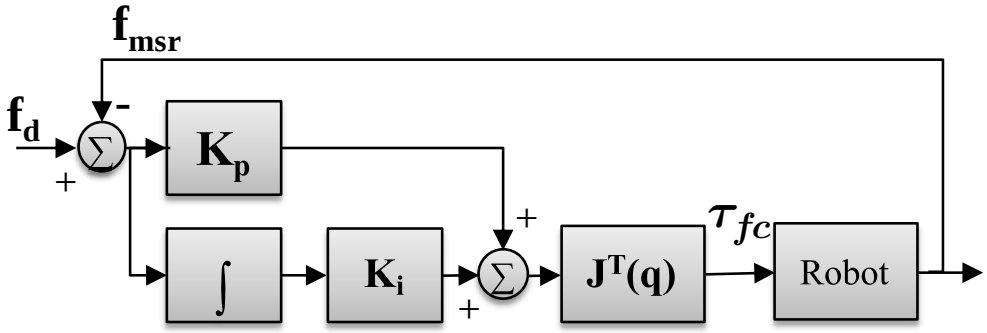



**Figure 4.10.:** Force Controller

ment. Extending the above equation 4.10 with signal clipping following is obtained as a new control law:

$$\boldsymbol{\tau_{fc}} = \boldsymbol{J^T(q)} \left( \boldsymbol{K_p e_f} + \boldsymbol{K_i} sat(\int \boldsymbol{e_{fe}} dt) \right) \tag{4.12}$$

$$\boldsymbol{e_f} = \boldsymbol{f_d} - \boldsymbol{f_{msr}} \; ; \boldsymbol{f} = \begin{bmatrix} force \\ torque \end{bmatrix}_{1x6} \tag{4.13}$$

**Hybrid Controller Implementation:**

The aforementioned two controllers are illustrated as implemented independently of each other. However, the torque command from both the controllers cannot be applied simultaneously on all dimensions. A selection mechanism for controller switching between position control and force control should be included in the hybrid position-force controller as a combined scheme to achieve compliant tasks. A simple classical hybrid control strategy proposed in [CR79] is shown in Fig.4.11. Control law for the Hybrid Controller shown in the figure 4.11 is written as follows:

$$\boldsymbol{\tau} = \boldsymbol{\tau_{ic}} + \boldsymbol{\tau_{fc}} \tag{4.14}$$

$$\begin{aligned} \boldsymbol{\tau} = \boldsymbol{J^T(q)}(\boldsymbol{K_c(S)}(\boldsymbol{e_p}) + \boldsymbol{D_x J(q)\dot{\theta}} + g(\boldsymbol{q})) \\ + \boldsymbol{J^T(q)}(\boldsymbol{K_p}(\widetilde{\boldsymbol{S}})(\boldsymbol{e_f}) + \boldsymbol{K_i}(\widetilde{\boldsymbol{S}}) \int \boldsymbol{e_f} dt) \end{aligned} \tag{4.15}$$

Here $\tau$ is Total Joint Torque, $\tau_{ic}$ is from Impedance Controller, from equation 4.8. $\tau_{fc}$ is from force controller, from equation 4.10. variable $S$ and $\widetilde{S}$ denotes the diagonal selection matrix and its complement. The selection matrix enables the selection of position control or force control in each dimension. This hybrid control scheme acts as a basis for control

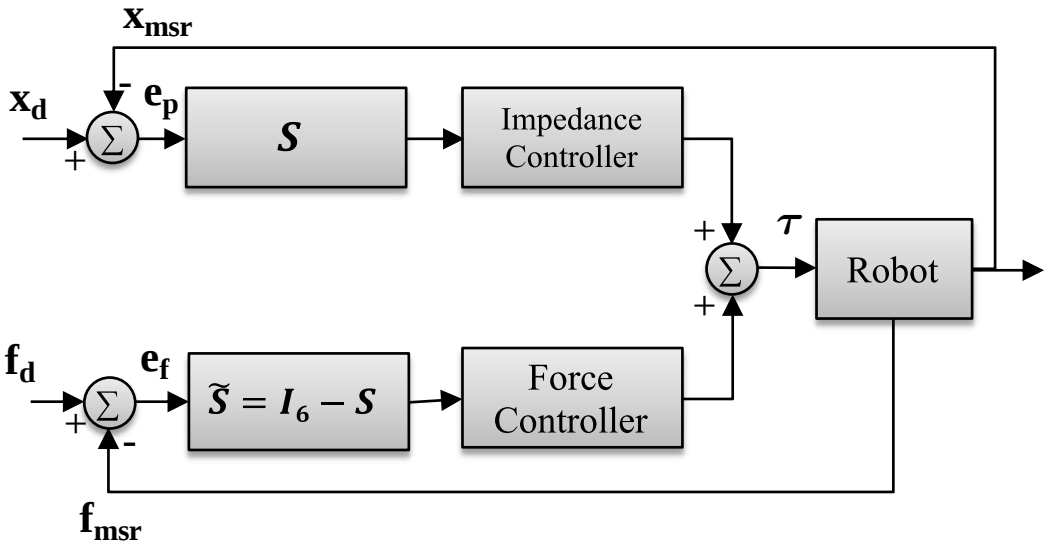



**Figure 4.11.:** Classical Hybrid Position-Force Controller

for compliant tasks in this thesis. Yet, the controller needs to be adapted for the skills and further modifications will be done based on skill parameterization in section 4.2.3.

## 4.2. Phase II: Skill Based Parameterization

In this phase, skill-specific parameters are identified for each skill in each module implemented in Phase-I. Each predefined skill needs to be parameterized and the parameters required in each module implemented in Phase I need to be identified. In this phase, demonstrated data is analyzed to identify and extract the skill-based features along with the identification and extraction of features specific to skills and adapted hybrid controller for skills. The overall workflow of Phase II is depicted in the figure 4.12.

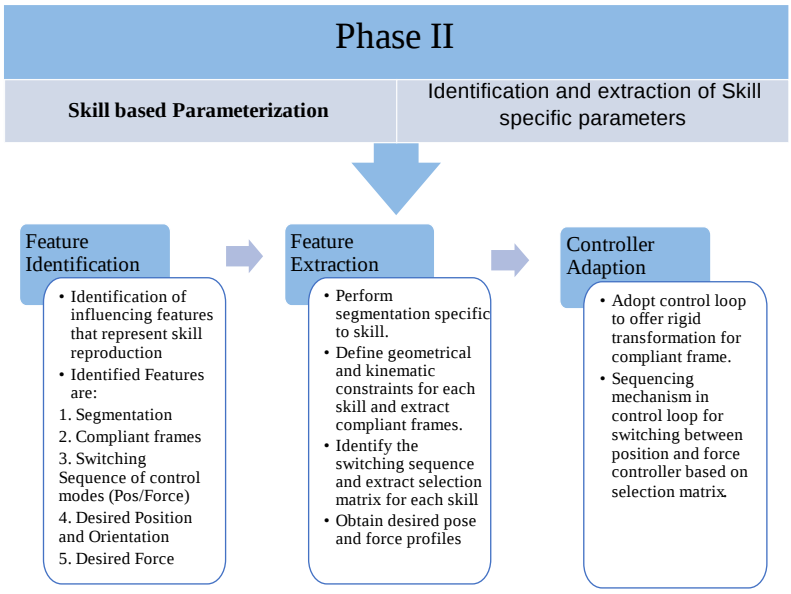



**Figure 4.12.:** Phase II

### 4.2.1. Feature Identification

Feature Identification is one of the major contributions of this thesis. Identifying features is a challenging task because features need to be identified only those that are relevant to represent each skill and reproduce the skill by learning. The main objective of the thesis is to develop a framework for the learning of contact-based skills. Therefore, features that are identified play a key role in learning a skill and reproduce it. Features are identified by analyzing the existing research under skill-based Learning from Demonstration context. By analyzing the related research works provided in section 3.3, it is noticeable that most common procedures to learn and reproduce the skills are similar to the approaches provided in Phase-I, such as segmentation, constraint extraction, DMP, and Hybrid controller. However, approaches employed to learn and reproduce the skill vary greatly based on their type of task and skill in their respective research. There is a need to identify such parameters that vary among the skills. Here, such important features

that vary among skills are presented as Trajectory Segments, Compliant Frames, Switching Sequence or Selection Matrix, and Trajectory Learning. These features should be extracted from demonstration data to learn and reproduce a skill. A detailed explanation for each feature and the reason identifying them as relevant features is as follow:

**1)Trajectory Segments:** Learning a trajectory can be done by learning DMPs of position, orientation and force profiles. However, complex trajectories are difficult for approximation and hard to represent within one Movement Primitive as explained in 4.1.1. Segmentation needs to be performed before learning trajectories. Segmentation is one of the most common steps observed in many research works prior to MP learning. However, the segmentation procedure varies based on the task considered. In contrast to the other research works presented in section 3.3, the aim is to identify what type of segmentation should be employed for each skill. Hence, segmentation is considered as one of the features of the skill.

**2)Compliant Frames:** Compliant frames are local coordinate systems where contact forces can be intuitively applied and are different for different skills, which are also called constraint frames as mentioned above. The force application cannot be generalized for all skills. For instance, for a sliding skill, the constant normal force needs to be applied on to the environment, whereas for contouring skill, the varying force needs to be applied. Every skill has a unique way to deal with force application and such compliant frames vary for every skill. Identifying the best possible way to deal with force in each skill is essential to learn and reproduce it effectively. Hence, Compliant frames are considered as one of the important features to be extracted for each skill, respectively.

**3)Selection Matrix:** Selection matrix is a diagonal matrix and a control parameter of hybrid controller that defines the control strategy between position and force control in each direction in the compliant frame. Each skill has a unique way to deal with force application and therefore Selection matrix that defines the control strategy is also unique to each skill. However, defining a selection matrix depends on the extracted compliant frames. In the subsequent section, defining a selection matrix for each skill based on the requirements of interaction forces and compliant frames is illustrated.

**4) Desired Trajectories:** Desired Position, orientation, and Force profiles by learning DMP or extracting from data. Desired Contact forces that should be applied on the environment vary from skill to skill. Desired forces are extracted from the demonstrated data for every task. However, the extraction procedure varies between skills which is based on the requirements of force interaction. An intuitive way of force application is defined on compliant frames. Varying Force profiles required for certain skills like contouring can also be learned using DMPs similar to position and orientation trajectories.

### 4.2.2. Feature Extraction

Skill-based features identified in the previous section need to be extracted for every specific skill which is predefined. Feature extraction procedures vary among the skills and careful methods need to be employed for extraction. Feature extraction is also a major contribution of this thesis to achieve the objective for the learning of contact-based skills. This section covers the extraction of identified features for each skill. As a part of this thesis, four basic skills such as 1)Slide, 2)Contour 3)Touch, and 4)Press are chosen as examples and further analysis and experiments are carried out on these chosen skills. Skill definition and extraction of each feature of skill are explained subsequently for each example defined as follows:

#### Skill 1: Slide

Sliding skill is defined as a skill to perform sliding operation on an arbitrary plane in 3D space. A constant force should be applied normal to the plane throughout the trajectory. In order to learn and reproduce sliding skill, the robot should learn the trajectories of motion and contact normal force and its application on the environment as shown in Fig.4.13, which can be achieved by learning the identified features.

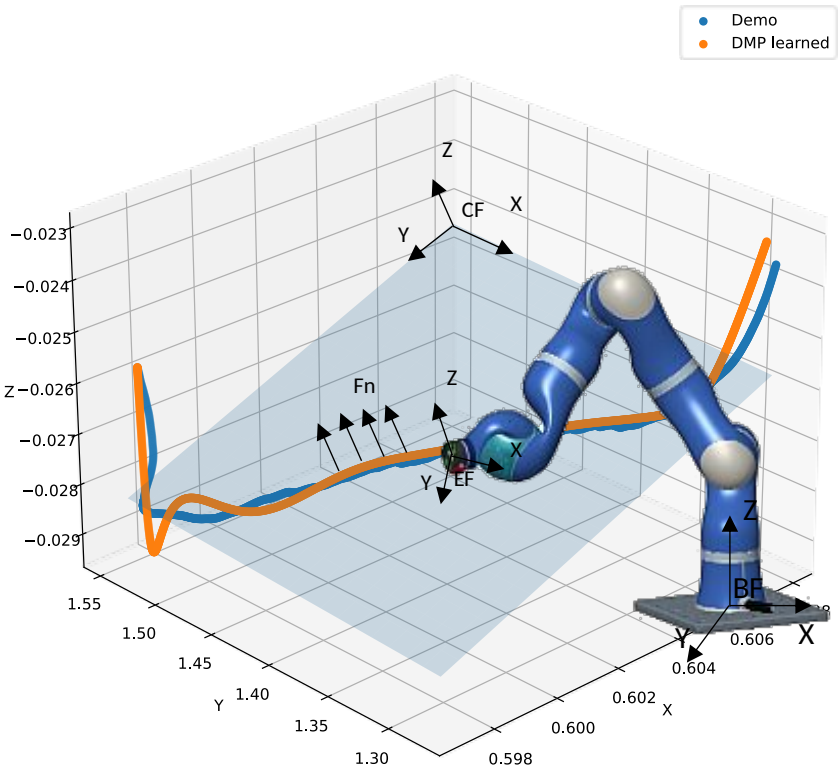



**Figure 4.13.:** Illustration of Skill 1: Slide

*1)Segmentation:* For the Sliding skill, Unsupervised cluster-based segmentation is employed to segment trajectories into multiple movement primitives by considering both position and force data of demo trajectory. Unsupervised cluster-based segmentation is considered here because the trajectory needs to be subdivided into segments that possess different motion parameters. For example, Compliant plane fitting cannot be fit on the entire trajectory as the application of force may vary at different parts of the long trajectory. This motion behavior can be easily grouped into smaller segments using cluster-based segmentation.

*2)Compliant Frames:* Considering Sliding skill, a constant normal force needs to be applied on a plane of a trajectory. Hence, a compliant frame can be defined on the plane, and such forces can be applied only on the Z-axis of the compliant frame. In order to define such a compliant frame, a plane needs to be fit on the segment. Here, the constraint extraction module, which is explained in the previous section, is used to fit planar constraint to extract plane.

*3)Selection Matrix:* In order to reproduce demonstrated sliding skill, the selection matrix here is fixed with respect to the compliant frame as it is required to apply a constant force normal to the plane and the compliant frame is defined on a plane. Now with respect to the CF, force is applied only in Z direction throughout the segment. Hence, position control is selected in X and Y direction, and force control is selected in the Z direction. It should be remembered that the selection matrix defined here is with respect to CF. This gives us a basic understanding that how important to define CF to have an intuitive understanding of force application.

*4)Desired Position:* Desired positions for slide skills are learned from DMP module. Change in initial and goal points can be adapted easily for the segments.

*5) Desired Orientation:* Orientations for slide skills are similar to position quantities. Thus, orientation is learned as quaternions using the Quaternion DMP module.

*6) Desired Forces:* from the skill definition for slide skill, a constant force is required to be applied in the direction of the plane normal or in the direction of the Z-axis to the CF. Therefore, the desired force is constant throughout for each segment and needs to be extracted from the demonstrated data. A simple average of demo forces in the Z direction after transformation into CF from BF is not a viable solution for extracting constant forces. Since the forces are not constant due to uncertainty and noise in data during the demonstration process. Additional care needs to be taken to extract the constant force in the Z direction, as shown in Fig.4.14. The figure shows that the constant region of force profile is limited to a certain range. Extracting the average value for the entire segment is not suggested and needs to be extracted by averaging the values of the constant

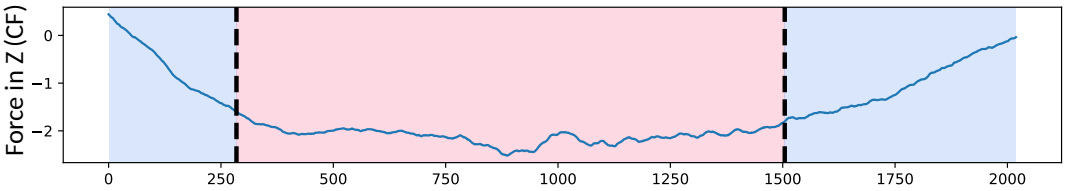



**Figure 4.14.:** Constant Force extraction from low variance region

region. The force profile is segmented again using change point segmentation to obtain segments of similar values and obtain desired force by averaging the region of low variance to address this issue. Thus, a viable constant value can be obtained by averaging a low variance region of the force profile of the entire segment.

The summary of the feature extraction for the Slide skill is shown in table 4.1.

| S.No | Feature | Description |
|---|---|---|
| 1 | Segmentation Type | Demonstrated data is segmented into multiple segments using Unsupervised clustering methods, each segment represents a movement primitive. |
| 2 | Compliant Frame | Obtain CF for each segment by fitting a "plane" constraint. |
| 3 | Selection Matrix | Fixed with respect to CF. Position control in X and Y directions and force control in Z direction with respect to CF. Selection Matrix = diag([1,1,0,1,1,1]). |
| 4 | Desired Position | DMP in X,Y dimensions from demo data |
| 5 | Desired Orientation | Quaternion DMP. |
| 6 | Desired Force | Constant obtained from average of values in low variance region. |

**Table 4.1.:** Overview of Features Extraction of Skill 1: Slide

### Skill 2: Contour

Contouring skill performs a contouring task, in which varying force needs to be applied in the direction of interaction. In order to learn and reproduce contouring skill, force profile needs to be learned and varying compliant frame needs to be extracted. A local coordinate system based on the force applied throughout the trajectory is extracted at each point, shown in Fig.4.15, which can be defined as a compliant frame.

*1) Segmentation:* Similar to sliding skill, defining single MP for the entire trajectory is not a viable solution. Therefore, trajectory is segmented into multiple movement primitives by using Unsupervised cluster based segmentation based on position and force data from the demo trajectory.

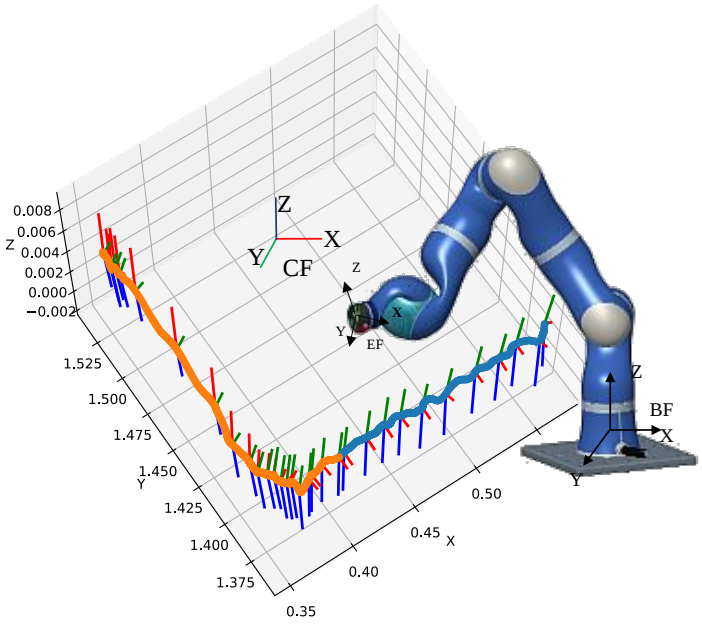



**Figure 4.15.:** Illustration of Skill 2: Contour

*2) Compliant Frame:* For contouring skill, CF varies dynamically at each point of the segment. CF is defined at a local coordinate system which is extracted based on demo data at each point. To extract local coordinates at each point, different methods can be used, for instance in [SLG18], local coordinates are extracted as Frenet-Serret frames (Tangent, Normal and Bi-normal orthogonal vectors) using Gram-Schmidt process at each point of the segment. This method of extracting local frames is quite convincing to define CF. However, the direction of forces is neglected and it is uncertain how forces are applied intuitively. A compliant way of defining constraint frames is proposed by Conkey et.al in [CH18]. In [CH18], Z-axis of the Compliant Frame at each point is aligned to the normalized force vector of unit length. And other orthogonal axes is constructed by aligning end-effector Y-axis as one orthogonal candidate and applied cross products to create a full right handed coordinate system. This method is sufficient to define CF, with which forces can be applied intuitively with respect to CF at each point. Although this method ensures compliant capabilities, the aforementioned method for construction of other orthogonals is not viable to be adapted in this thesis. Thus, only the idea of aligning Z-axis to normalized force vector of unit length is considered and the construction of another orthogonal axis can be done with efficient strategies. Tomas et.al in [TJM99] provided an efficient method for constructing a frame in the form of a rotation matrix that rotates a unit vector into another unit vector. Here, a unit vector of Z axis in CF is rotated into another unit vector of normalized forces and from the method proposed in [TJM99], a rotation matrix will be constructed, which defines a compliant frame at that point.

*3)Selection Matrix:* In order to reproduce demonstrated contouring skill, Selection matrix here is fixed with respect to the compliant frame. With respect to the CF at each point, force is applied only in Z direction throughout the segment. Hence, position control is selected in X and Y direction and force control is selected in the Z direction. It should be remembered that the selection matrix defined here is with respect to CF and is defined intuitively.

*4)Desired Position:* Desired positions for slide skills are learned from DMP module. Change in initial and goal points can be adapted easily for the segments.

*5) Desired Orientation:* Orientations for contour skills are similar to position quantities. Thus, orientation is learned as quaternions using Quaternion DMP module.

*6) Desired Forces:* Desired forces can be defined in two methods for contouring skill based on task requirements. A simple way of defining desired force is by fixing a constant value to apply on the contact surface in the direction of Z-Axis of learned CFs constructed earlier. Another way of defining forces is by extracting magnitude of normalized force vector at each point of the trajectory. The force profiles are learned by learning a DMP over force data and the magnitude of the normalized forces at each point can be extracted such that varying force applications at varying contact surfaces can also be incorporated.

The summary of the feature extraction for the Contour skill is shown in table 4.2.

| **S.No** | **Feature** | **Description** |
|---|---|---|
| 1 | Segmentation Type | Demonstrated data is segmented into multiple segments using Unsupervised clustering methods, each segment represents a movement primitive. |
| 2 | Compliant Frame | Local Coordinate system at each point with Z axis aligned to Normalized force vector. |
| 3 | Selection Matrix | Fixed with respect to CF. Position control in X and Y directions and force control in Z direction with respect to CF. Selection Matrix = diag([1,1,0,1,1,1]) |
| 4 | Desired Position | DMP in X,Y dimensions from demo data |
| 5 | Desired Orientation | Quaternion DMP |
| 6 | Desired Force | Trajectory of force obtained by computing Magnitude of Normalized force vector at each point of trajectory. |

**Table 4.2.:** Overview of Features Extraction of Skill 2: Contour

### Skill 3: Touch

Touch skill is to interact with the environment until the robot slightly touches the contact point. Force is not applied on the environment but force measured during the demonstration for touching an environment will be adapted as a goal force to perform touch operation. When a touch skill is demonstrated, robot needs to learn path and contact point from the demonstrated data and should be able to reproduce and adapt to the dynamics of the environment. After demonstrating touch skill, during reproduction, the robot moves along the learned path until the contact point has reached and then returns in the other learned path after contact point. While executing a touch skill, three different situations might occur during interaction with the environment as shown in Fig.4.16.

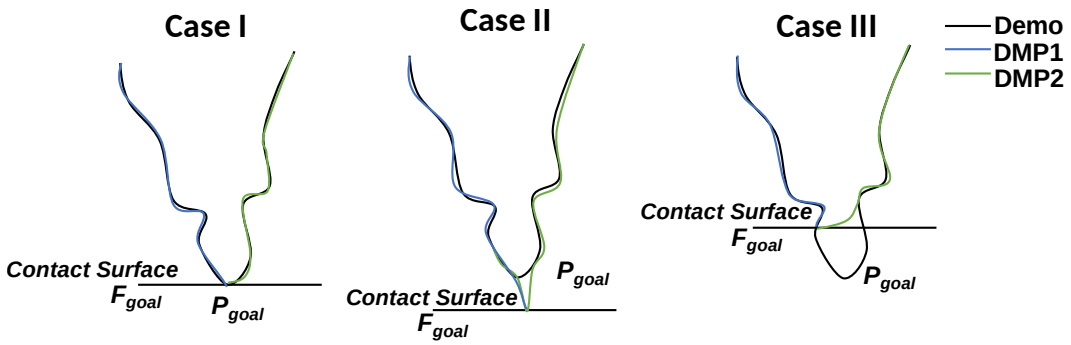



**Figure 4.16.:** Illustration of Robot Interaction with environment

In case I, dynamics of the environment may not change, such that goal point and contact point remain same and touch skill will be executed as per demonstration. In case II, during execution, when the environment moves farther, the contact point may not coincide with the goal point and such a goal point needs to be changed dynamically to reach the contact point. In case III, in contrast to case II, if contact point reaches earlier than expected, the movement to reach goal point should terminate and should switch to another learned path with current location as new initial point.

*1) Segmentation:* In order to learn and reproduce touch skill, the trajectory should be segmented into two paths, approach path and return path. The segmentation is done at the point of contact with the environment. After analyzing demonstration data, contact point usually occurs at a point where maximum force occurs. Hence, a simple method of segmentation is done at maximum force point in the demo trajectory. Other methods of segmentation from the previous section like cluster based and change point method based segmentation also gives similar results, yet simple method of segmentation is more intuitive if dynamics of the environment are maintained.

*2) Constraint Frame:* While executing touch skill, robot interacts with an environment only near contact point. Compliant Frame needs to be extracted at this contact point to

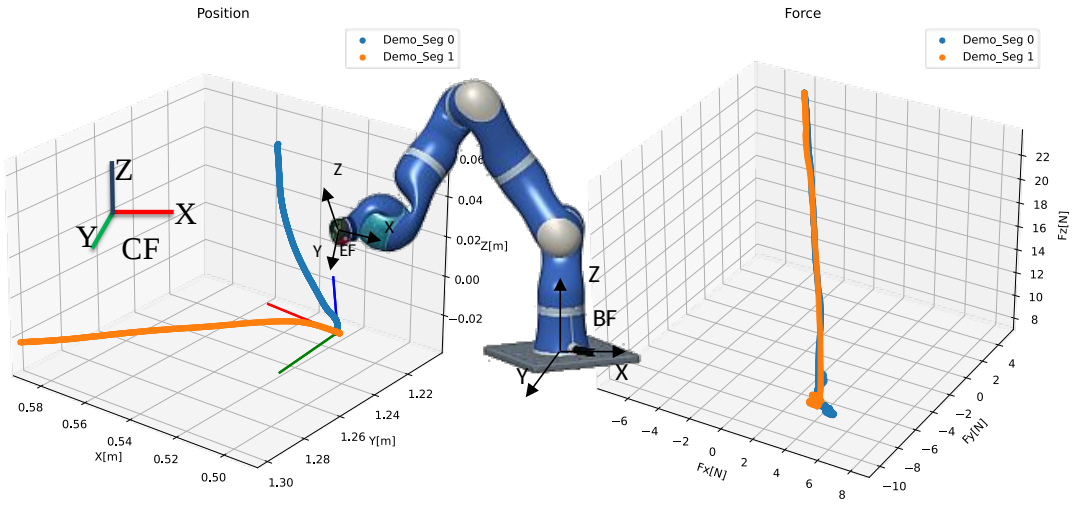



**Figure 4.17.:** Illustration of Skill 3: Touch

track the forces for the identification of contact state. A local coordinate system similar to contour skill is extracted at this point, such that instead of tracking forces in all directions to identify contact state, force is tracked along the direction of Z-axis in CF.

*3) Selection Matrix:* For touch skill, the interaction forces are not reproduced but tracked to reach the contact point or touch location. Thus, only position control is sufficient to move the robot and force control is not required in any direction. Forces are tracked simultaneously during the movement learned from demonstration only to track contact state and force control switching is not required while executing touch skill.

*4) Desired Position:* DMPs are used to learn the path of approach and return. However, the initial and goal points are adapted to the changes in the contact state while reproducing as shown in the figure 4.16 by considering the feedback from the robot.

*5) Desired Orientation:* Orientations for touch skills are similar to position quantities. Thus orientation is learned as quaternions using Quaternion DMP module.

*6) Desired Forces:* Although force control is not used to perform touch skill, goal force defined as force at contact point needs to be tracked in the direction of Z-axis in CF to identify contact point. Thus, the desired force to track the contact point can be obtained by computing the normalized force vector at the segmentation point, which is also a contact point.

The summary of the feature extraction for the Touch skill is shown in table 4.3.

| S.No | Feature | Description |
|---|---|---|
| 1 | Segmentation Type | Segment the demo data into two MPs at a point where maximum force occurs. |
| 2 | Compliant Frame | Extract Local Coordinate system at each point with Z axis aligned to Normalized force vector. |
| 3 | Selection Matrix | Position control in X .Y and Z directions, until the goal point has reached, Selection Matrix = diag([1,1,1,1,1,1]), Switch to Force control after goal point reached, Selection Matrix = diag([0,0,0,1,1,1]). Switch Back to position control to retract after reaching goal force, Selection Matrix = diag([1,1,1,1,1,1]) |
| 4 | Desired Position | DMP in X,Y,Z dimensions from demo data |
| 5 | Desired Orientation | Quaternion DMP |
| 6 | Desired Force | Normalized force vector at the segmentation point which is touch point. |

**Table 4.3.:** Overview of Features Extraction of Skill 3: Touch

**Skill4: Press**

Press skill is similar to touch skill except that force control is involved here when the robot interacts with the environment. In contrast to touch skill, in press skill, the robot applies forces learned during a demonstration on the environment after reaching the contact point. Until contact point is reached, robot moves in path learned from demonstration and robot returns from the contact environment after applying learned forces. The entire process of executing Press skill is shown in Fig.4.18 and Fig.4.19.

*1) Segmentation:* From the figure 4.18 it is certain that the trajectory of press skill needs to be segmented into three parts. The first segment is an approach path to reach contact with the environment. The second segment represents the interaction phase, where robot interacts with the environment by applying contact forces learned from demonstration. And the third segment is return path away from the contact with the environment. The pattern of segments varies quite convincingly through time. Such disruptive event changes that can reflect on the position and force variables of demonstrated data are noticeably identifiable using change point detection based segmentation.

*2) Constraint Frame:* Similar to touch skill, compliant frames are extracted only at segmentation points which are also interaction points. Compliant Frame needs to be extracted at this contact point to track the forces for the identification of contact state. A local coordinate system similar to contour skill is extracted at this point, such that instead of tracking forces in all directions to identify contact state, force is tracked along the Z-axis in CF.

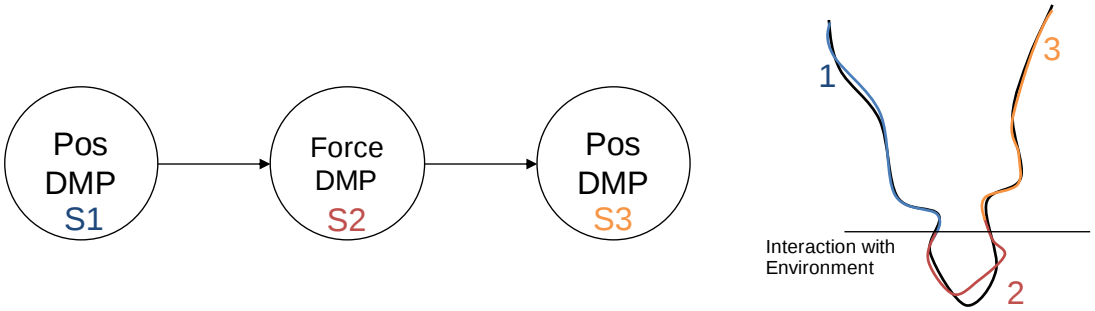



**Figure 4.18.:** Illustration of Robot Interaction with environment

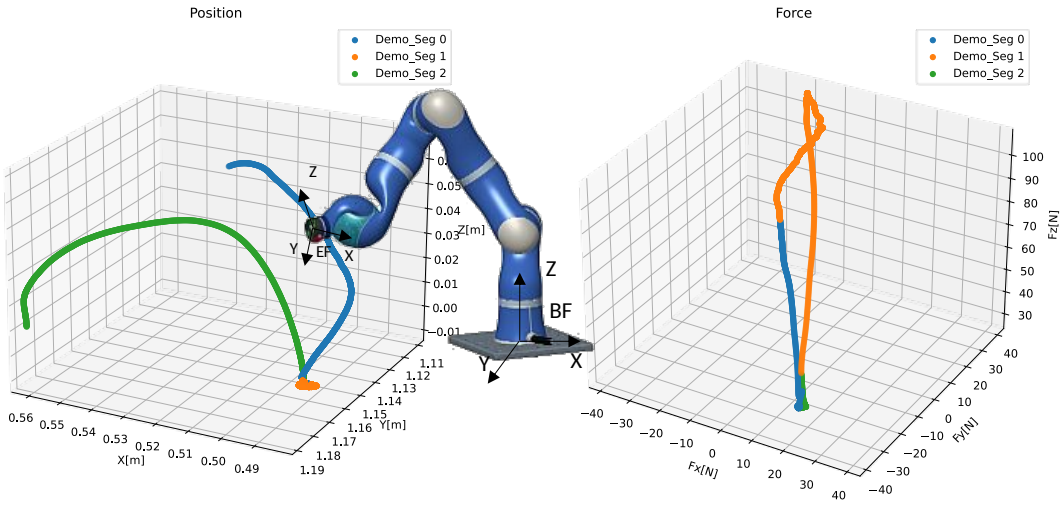



**Figure 4.19.:** Illustration of Skill 4: Press

*3) Selection Matrix:* For press skill, during approach path the interaction forces are only tracked to reach the contact point or interaction point. Thus, only position control is sufficient to move the robot and force control is not required. Selection matrix is chosen such that X, Y, and Z are chosen for position control. While executing the second segment, robot is under interaction with the environment and should be switch to the force controller. After applying learned interaction forces, control will switch back to position control while executing the third segment.

*4) Desired Position:* DMPs are used to learn the path of approach and return. However, the initial and goal points are adapted to the changes in the contact state while reproducing as shown in the figure 4.18 by considering the feedback from the robot.

*5) Desired Orientation:* Similar to position learning, orientations are also learned using Quaternion DMPs to adapt the dynamic changes in the environment.

*6) Desired Forces:* Learning press skill involves learning two different force components. Firstly, contact force at interaction points or segmentation points similar to touch skill needs to be learned to track the contact state during the approach path. Additionally, during the interaction phase, the robot should also learn the forces applied during the

demonstration while in contact. Hence, force profiles during the second segment can be learned by using DMPs. The summary of the feature extraction for the Press skill is shown in table 4.4.

| S.No | Feature | Description |
|---|---|---|
| 1 | Segmentation Type | Segment the demo data into three MPs from change point detection algorithm. |
| 2 | Compliant Frame | Extract Local Coordinate system at each point with Z axis aligned to Normalized force vector. |
| 3 | Selection Matrix | Position control in X .Y and Z directions, until the end of first segment, Selection Matrix = diag([1,1,1,1,1,1]), Switch to Force control during second MP, Selection Matrix = diag([0,0,0,1,1,1]). Switch Back to position control to retract after during third MP, Selection Matrix = diag([1,1,1,1,1,1]) |
| 4 | Desired Position | DMP in X,Y,Z dimensions from demo data |
| 5 | Desired Orientation | Quaternion DMP |
| 6 | Desired Force | Force profile in X,Y,Z for second segment by learning DMPs |

**Table 4.4.:** Overview of Features Extraction of Skill 4: Press

### 4.2.3. Skill Based Controller Adaption

Classical Hybrid controller from earlier sections shown in Fig.4.11 is not adequate to reproduce skills that are learned based on the aforementioned features. The existing controller works well if there are no transformations of state variables. However, Compliant Frames are one of the major features of the skill that undergo transformation (Rotation and Translational), which favours to apply forces intuitively. Consequently, Stiffness parameter $K_c$ and selection matrix $S$ in the equation 4.14 are diagonal matrices framed based on interaction forces are also defined with respect to the compliant frame. CF varies for each skill and such transformations should be incorporated in the control law written in equation 4.14.

Conkey et.al in [CL18] addressed the similar issue. Selection Matrix is pre-multiplied with the transpose of the Rotation matrix which defines CF and post multiplied with the same Rotation matrix of CF. This idea of transformation of matrices can be employed in this thesis to be able to incorporate in the Hybrid control law. However, the same strategy cannot be adapted directly without further modifications for two main reasons: 1) Stiffness matrix is not defined in [CH18] because the Cartesian inverse dynamic controller is used instead of impedance controller and 2) Only rotational transformation of Constraint

frames where considered which is sufficient for the tasks explained in the paper, whereas in this thesis the objective is to develop a robust and generic controller that should be able to use for different skills that includes transformations considering both rotation and translation of compliant frames.

A unified position force control is proposed by Marin et.al in [MW16] is an another control strategy that allows rigid transformation of the compliant frames. The motion commands are represented by means of twists and wrenches and are kinestatically filtered before passing to respective position and force controllers. The filters are designed by considering the rigid transformation of compliant frame such that by filtering, the transformation of commands is handled by default. This is an appealing strategy to be incorporated in this thesis. However, this methods cannot be adopted directly because the idea of addressing the rigid transformation of CF by Kinestatic filtering performed on twist and wrench commands before passing it into respective controllers is different with respect to this thesis. Instead of transforming commands, it is intended to transform only selection matrix and stiffness matrix because motion and force commands are learned through feature extraction methods provided earlier are in base frame. Also, in contrast to the requirement of defining stiffness matrix in CF, kinestatic filtering method according to [MW16] has not addressed the transformation of stiffness matrix because it is fixed with respect to the base frame.

Based on the understanding acquired from the above two strategies employed in [CL18] and [MW16] and the requirements of this thesis for handling the rigid transformations of compliant frames, a generic control equation is derived suitable for most of the skills as follows. Rewriting the classical hybrid control law from equation 4.14:

$$\boldsymbol{\tau} = \boldsymbol{J^T}(\boldsymbol{q})(\boldsymbol{K_c}(\boldsymbol{S})(\boldsymbol{e_p}) + \boldsymbol{D_x}\boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{\theta}} + g(\boldsymbol{q})) + \boldsymbol{J^T}(\boldsymbol{q})(\boldsymbol{K_p}(\widetilde{\boldsymbol{S}})(\boldsymbol{e_f}) + \boldsymbol{K_i}(\widetilde{\boldsymbol{S}}) \int \boldsymbol{e_f} dt) \tag{4.16}$$

for simplicity, considering orientations are represented as euler angles and rewriting above equation in End-effector frame as:

$$\begin{aligned} \boldsymbol{\tau} = \boldsymbol{J^T}(\boldsymbol{q})(^{\boldsymbol{EF}}\boldsymbol{K_c}(^{\boldsymbol{EF}}\boldsymbol{S})(^{\boldsymbol{EF}}\boldsymbol{e_p}) + \boldsymbol{D_x}\boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{\theta}} + g(\boldsymbol{q})) \\ +\boldsymbol{J^T}(\boldsymbol{q})(\boldsymbol{K_p}(^{\boldsymbol{EF}}\widetilde{\boldsymbol{S}})(^{\boldsymbol{EF}}\boldsymbol{e_f}) + \boldsymbol{K_i}(^{\boldsymbol{EF}}\widetilde{\boldsymbol{S}}) \overset{\boldsymbol{EF}}{\int} \boldsymbol{e_f} dt) \end{aligned} \tag{4.17}$$

In the above equation stiffness $\boldsymbol{K_c}$ and selection matrix $\boldsymbol{S}$ and its compliment $\widetilde{\boldsymbol{S}} = (\boldsymbol{I_6} - \boldsymbol{S})$ are mentioned in end effector frame $\boldsymbol{EF}$. Aforementioned requirement for Stiffness and Selection matrices as diagonal matrices should be defined in compliant frame. Representing above quantities from EF to CF needs transformation of 6x6 matrices between one

frame to another frame. The transformation of Stiffness matrices between frames is provided in [Lon87] and explained in detail in [MLS17]. The transformation of 6x6 matrices can be done by using adjoint matrices derived from homogeneous transformation matrix of a transformed reference frame. In contrast to homogeneous transformation, adjoint matrices are used to transform twists and wrench quantities represented in 6x1 vectors respectively.

CF can be represented in homogeneous transformation matrix with rotation R and translation p as:

$$^{EF}\boldsymbol{T}_{\boldsymbol{CF}} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{p} \\ 0 & 1 \end{bmatrix} \tag{4.18}$$

According to [MLS17], the adjoint matrices for twist and wrench quantities from the above homogeneous transformation 4.18 can be written as follows:

$$^{EF}\boldsymbol{Ad}_{\boldsymbol{CF}} = \begin{bmatrix} \boldsymbol{R} & [\hat{\boldsymbol{p}}]\,\boldsymbol{R} \\ 0 & \boldsymbol{R} \end{bmatrix} \quad and \quad ^{EF}\boldsymbol{Adg}_{\boldsymbol{CF}} = \begin{bmatrix} \boldsymbol{R} & 0 \\ [\hat{\boldsymbol{p}}]\,\boldsymbol{R} & \boldsymbol{R} \end{bmatrix} \tag{4.19}$$

Where $^{EF}\boldsymbol{Ad}_{\boldsymbol{CF}}$ is used for transformation of twist and $^{EF}\boldsymbol{Adg}_{\boldsymbol{CF}}$ is used for transformation of wrench vectors. And $[\hat{\boldsymbol{p}}]$ is skew symmetric matrix of the translational vector $\boldsymbol{p}$. Using above adjoint matrices from equation 4.19, transformation of stiffness and selection $\boldsymbol{K_cS}$ and compliment of selection matrix $\widetilde{\boldsymbol{S}}$ can be represented in CF from EF as follows:

$$\boldsymbol{\Omega} \;=\; ^{EF}\boldsymbol{K}_{\boldsymbol{c}}\;^{EF}\boldsymbol{S} \;=\; ^{EF}\boldsymbol{Adg}_{\boldsymbol{CF}}\;^{CF}\boldsymbol{K}_{\boldsymbol{c}}\;^{CF}\boldsymbol{S}\;^{EF}\boldsymbol{Ad}_{\boldsymbol{CF}}{}^{-1} \tag{4.20}$$

$$\widetilde{\boldsymbol{\Omega}} \;=\; ^{EF}\widetilde{\boldsymbol{S}} \;=\; ^{EF}\boldsymbol{Adg}_{\boldsymbol{CF}}\;^{CF}\widetilde{\boldsymbol{S}}\;^{EF}\boldsymbol{Adg}_{\boldsymbol{CF}}{}^{-1} \tag{4.21}$$

By substituting equation 4.20 in 4.17, a new control law is obtained as follows:

$$\boldsymbol{\tau} = \boldsymbol{J}^{\boldsymbol{T}}(\boldsymbol{q})\left(\boldsymbol{\Omega}\left(^{EF}\boldsymbol{e}_{\boldsymbol{p}}\right) + \boldsymbol{D}_x\boldsymbol{J}(\boldsymbol{q})\,\dot{\boldsymbol{\theta}} + g(\boldsymbol{q}) + (\boldsymbol{K}_p\widetilde{\boldsymbol{\Omega}}\;^{EF}\boldsymbol{ef} + \boldsymbol{K}_i\int\widetilde{\boldsymbol{\Omega}}\;^{EF}\boldsymbol{e}_{\boldsymbol{f}}dt)\right) \tag{4.22}$$

Modified controller scheme based on equation 4.22 is shown in the figure 4.20

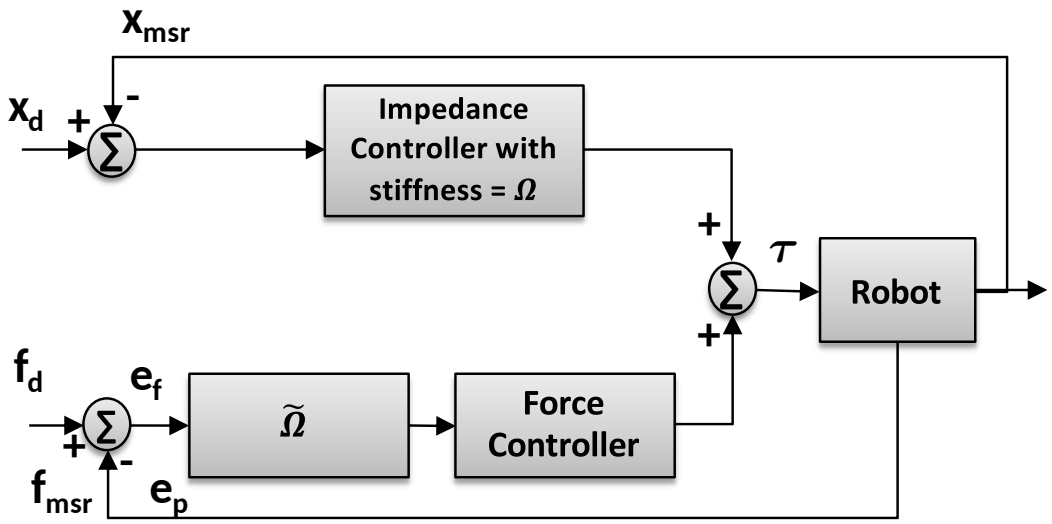



**Figure 4.20.:** Modified Hybrid Position-Force Controller

## 4.3. Summary

The presented approach of skill based learning for contact based tasks allows learning of demonstrated trajectory of a skill with just a single demonstration. The approach is employed in two phases. In the first phase, suitable segmentation algorithms for skill based learning is identified. Then on the segmented trajectory, to learn a Movement Primitive for each segment, Dynamic Movement Primitive is employed for learning of position, orientation and force quantities that facilitates adaption to novel situations such as new initial and final goal states. A suitable controller is identified which is a Hybrid Position-Force controller that allows control of position and force to execute contact based tasks. In the second phase, methods are employed focusing on skill based parameterization, where skill features are identified and extracted that represent a skill and are able to be used for reproducing a skill by learning. Later, hybrid position-force controller is adapted to the requirements based on features such as rigid transformation of task frames. In the next chapter, the evaluation of the approach will be presented by conducting experiments for each skill.

# 5. Experiments and Evaluation

For evaluating proposed skill based learning framework, existing demonstration data that is recorded by kinesthetic demonstration on DLR LWR-IV is used. Experiments for four preliminary skills: *Touch, Press, Slide and Contour* are conducted under Simulation environment. This section details the experimental setup implemented for the approach employed, experiments conducted for each skill, and interpretation of results in comparison with simple trajectory learning.

## 5.1. Experimental Setup

Skills are learned from demonstration data and features are extracted using Skill Learning module implemented in Python based on the approach provided earlier. Extracted skill features are used to generate the position, orientation and force trajectories. Hybrid Position-Force controller derived in section 4.2.3 is implemented entirely in MATLAB Simulink package. A Python program is written to interface with Bullet Physics simulation environment using PyBullet API that handles the commands to robot and provides feedback of the robot state. The entire architecture runs under PC with configuration: OpenSuse Linux 15.2, Intel Xeon 8 core CPU @ 3.60GHz and 32GiB RAM.

Generated trajectories and skill features are provided as an argument to the Simulink Model for Hybrid Controller that generates the joint torque commands for robot. DLR LWR-IV robot model is loaded into the environment which is defined in Unified Robot Description Format (URDF) file. The robot recieves the command from the Simulink model and provides robot state as feedback to Skill Learning module and Simulink Model to generate commands for the next time step. The overview of the signal flow between modules is shown in Fig.5.1 All the modules are interfaced through a middleware called Links and Nodes, which is a DLR framework that offers interface between distributed computing modules based on publisher-subscriber model. The overview of interface is shown in the Fig.5.2.

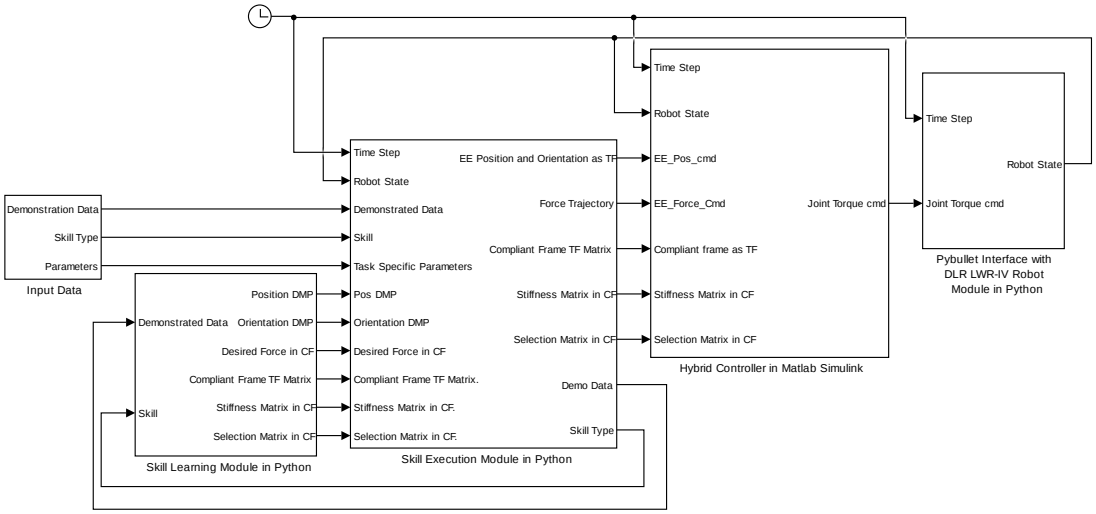



**Figure 5.1.:** Overview of the Experimental Setup

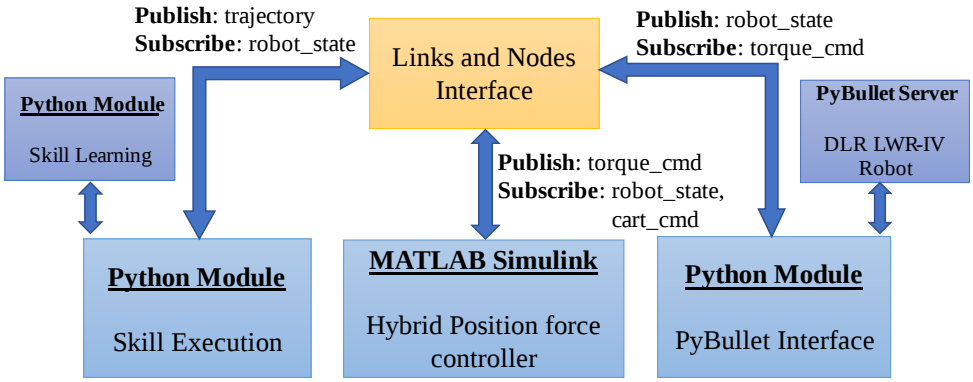
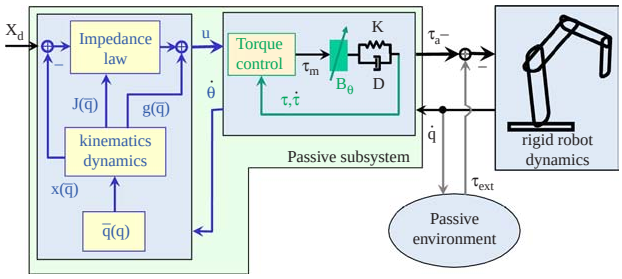


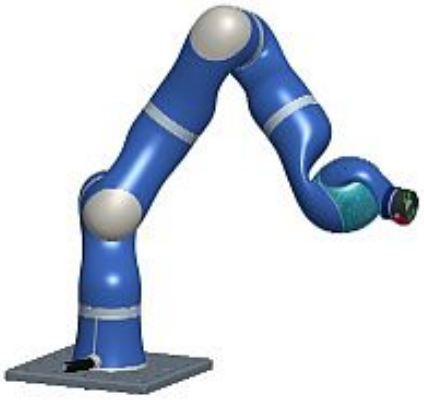

**Figure 5.2.:** Overview of the Interface

### 5.1.1. Demonstration Robot

Demonstrations are recorded on DLR LWR-IV robot by using Kinesthetic Demonstration strategy. LWR-IV is a Light Weight Robot as shown in Fig.5.3 has 7 DOF for manipulation with kinematics similar to human hand with force torque sensor mounted on the end effector joint. LWR Robot works under torque control mode at joint level enabling robot to control with joint torque commands. A cartesian Impedance Control at higher-level as shown in the Fig.5.3a [AHO07]. Joint torque control also enables the robot to control in "Zero Gravity Mode" in which the motor compensates the robot dynamics and weights. Demonstrations are carried out under Zero Gravity Mode and the demonstration data is recording with the help of force-torque sensor mounted on the robot 5.4. A high-level impedance control as explained in the section 4.1.4 generates torque commands for each joint by using robot kinematics, dynamics and current robot state for the commanded position. Joint torques generated by impedance controller are tracked and controlled by joint level torque controller for each joint enabling compliance behavior for the robot.



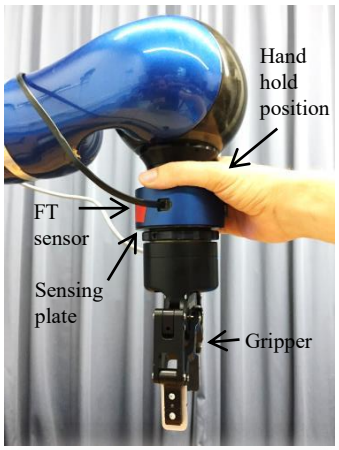

**(a)** Controller Overview of DLR LWR Robot



**(b)** Robot Model

**Figure 5.3.:** DLR LWR Robot, Source:[AHO07]

**(a)** Demonstrating a Skill




**(b)** Sensor mounting and hand hold position, Source:[ESL19]

**Figure 5.4.:** DLR LWR Robot, Source:[AHO07]

### 5.1.2. Links and Nodes

Links and Nodes (LN) is a realtime communication middleware, which is DLR's internal framework. It is similar to other frameworks like Robot Operating System (ROS) [OSR14] and Robot Construction Kit (RoCK) [Roc13]. It works on the basis subscriber-publisher model that interface different computing modules via topics or as services. Modules communicate with each other by publishing and subscribing the topics or service accordingly. Modules can either be deployed locally or via network. LN manager is a GUI tool that provides an overview of the interface and maintains the deployment of the modules, topics and interface independent of the system limitations. Modules can be configured and maintained through LN manager and also allows logging of published topics for debugging. Links and Nodes supports multiple platforms such as API are available for C++, Python and MATLAB Simulink to interface with LN for publishing and subscribing the

topics for interprocess communication. In this thesis modules in Python and MATLAB Simulink are implemented and interfaced through LN as shown in Fig.5.2. The modules of this thesis are maintained and deployed through LN manager as shown in Fig5.5

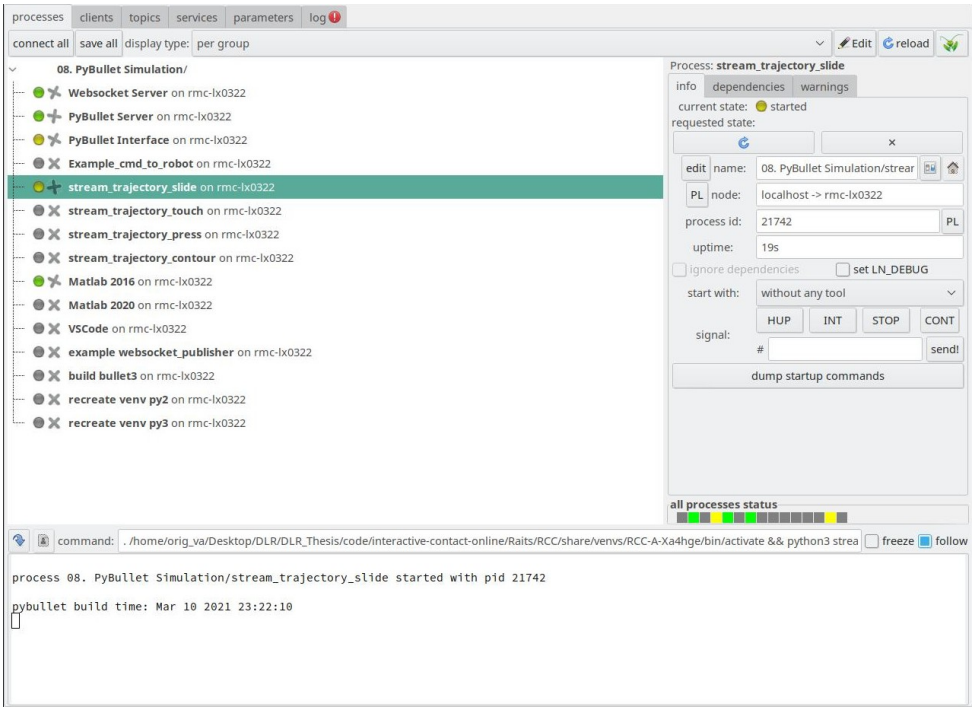



**Figure 5.5.:** Links and Nodes (LN Manager)

### 5.1.3. Skill Execution and Learning Module

Skill Execution module is a Python program that manages the execution of skill by importing data, skill learning, and generating commands to the hybrid controller. Module imports data which is recoded from the demonstrations and perform learning on the data using Skill Learning module. Learning a skill is extracting identified features such as position and orientation DMPs, compliant frames, selection matrix and desired forces specific to each skill as explained in section 4.2.2. Output of the Skill Learning module which is extracted features are used by the Skill Execution module to generate a cartesian command to the Hybrid Controller at each time step. The cartesian command generated is a trajectory of position, orientation and force profiles obtained from DMPs, adapted to the task requirements and current state along with other task specific parameters such as stiffness matrix, selection matrix and compliant frame.

Skill Execution Module is interfaced with other modules through Links and Nodes as shown in the Fig.5.2. Module publishes the cartesian command and subscribes to robot state.

### 5.1.4. Hybrid Controller Module

Hybrid Position-Force controller devised in the section 4.2.3 is realized in MATLAB Simulink package. The module subscribes to robot state from PyBullet Interface and cartesian command from Skill Execution Module. Joint torque commands for all joints of the robot are generated for each time step. This module publishes these commands as a topic to Links and Nodes Interface.

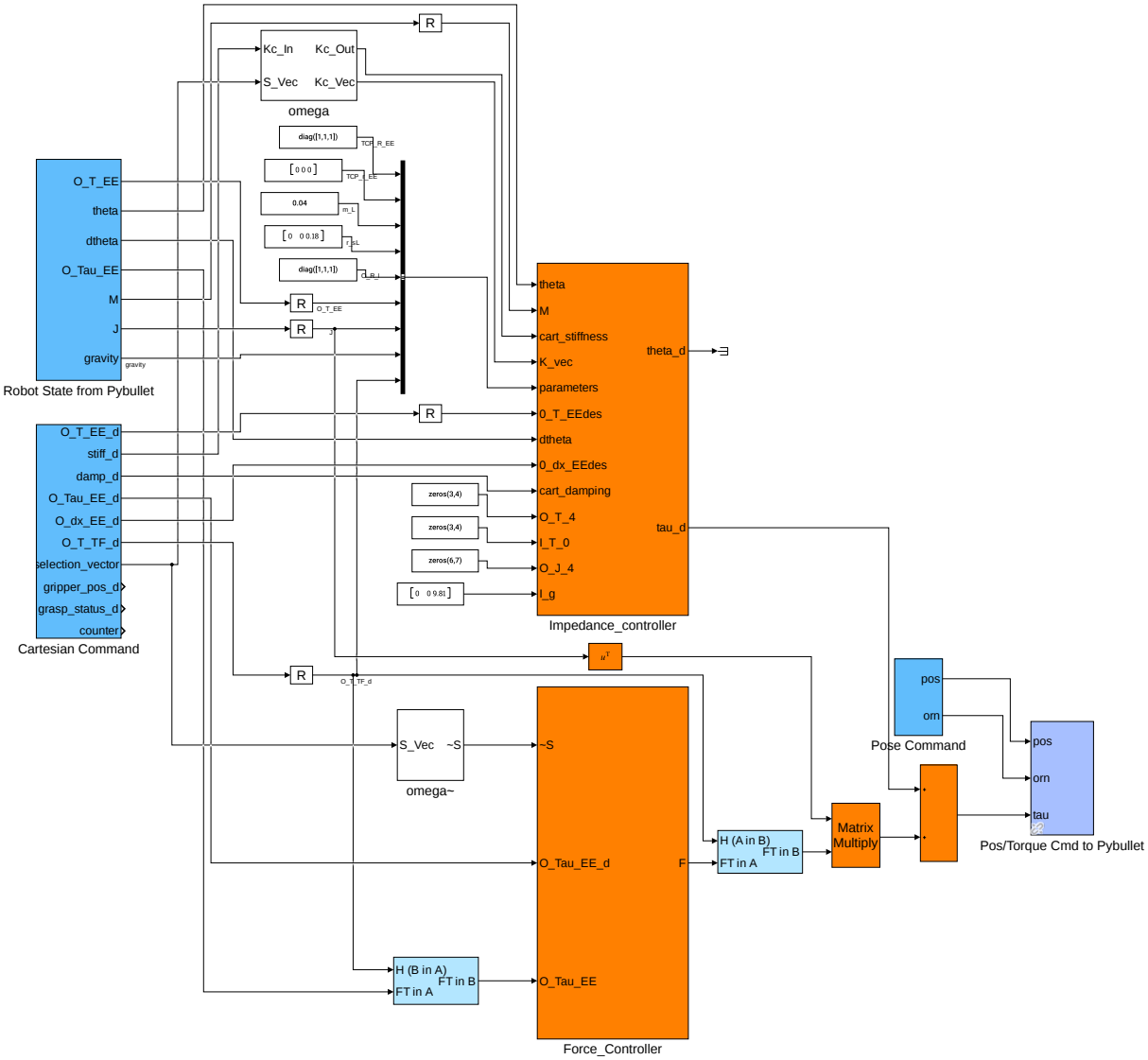



**Figure 5.6.:** Block Diagram of Hybrid Position-Force Controller in Simulink

### 5.1.5. PyBullet Interface Module

Experiments are carried out on simulation environment called Bullet Physics using PyBullet Python package. PyBullet is an easy to use Python module for physics simulation for robotics, games, visual effects and machine learning. PyBullet can load articulated bodies from URDF or SDF file formats. PyBullet provides forward dynamics simulation, inverse dynamics computation, forward and inverse kinematics, collision detection and ray intersection queries [E C16]. PyBullet Simulation Environment is chosen to conduct experiments because all the requirements such as simulation of kinematics and dynamic quantities, simulation with feedback of forces and torques, interaction and collision feedback and simulation of force torque sensors to conduct experiments for this thesis can be met by using PyBullet simulation package.

PyBullet Interface module is a Python code that interfaces with other modules of the project through Links and Nodes and manages the simulating robot and environment using PyBullet APIs. Robot model of DLR-LWR is defined in a URDF file included with meshes. This will be loaded into the environment along with other environment objects as shown in the Fig.5.7. The robot is configured to use in Torque Control mode with Force-Torque sensor enabled at end effector joint.

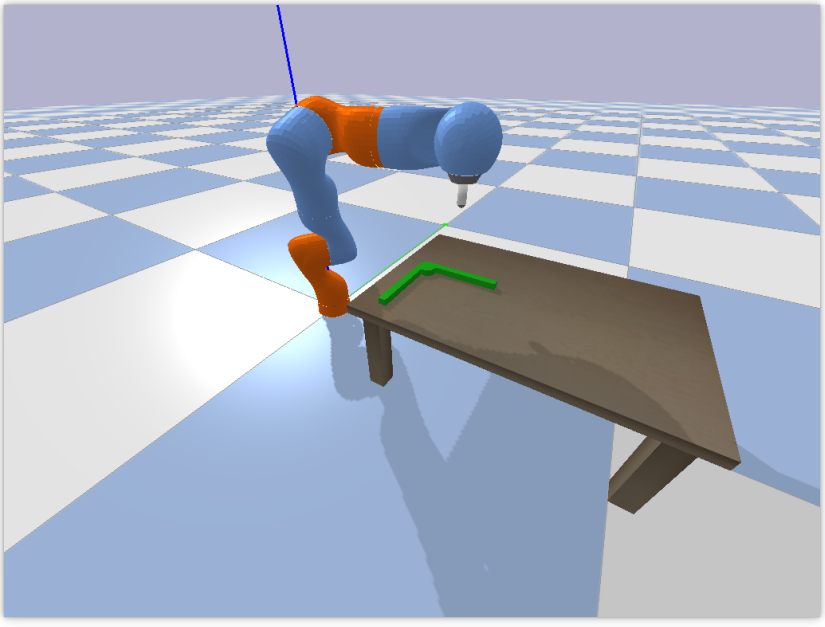


**Figure 5.7.:** PyBullet Simulation Environment with DLR LWR Robot

This module subscribes to the joint torque commands published by Hybrid controller of Simulink model and joint torques are commanded to the configured robot at each time step. Interface module publishes robot state that includes position, orientation, reaction forces, joint angles, mass matrix, Jacobian matrix and velocities at each time step.

## 5.2. Experiments

Aim of this section is to evaluate if the employed approach fulfills the goal of the thesis to learn contact based tasks by skill based learning approach with single demonstration. Experiments are conducted on four basic skills with multiple trials by changing the environment in each trial. Skills are learned from single demonstration and reproduced under simulation with varying environment. The results are extracted and evaluated for each skill as follows:

### 5.2.1. Skill 1: Slide

For a sliding skill, constant contact force normal to an arbitrary plane needs to be applied. Demo trajectory is segmented by using Agglomerative cluster based segmentation to reduce complexity of the trajectory for DMP learning. By fitting a plane constraint to each segment, compliant frame can be extracted. A constant normal force can be extracted from the low variance region of Z-axis force profile in compliant frame for each segment. Figure 5.8 shows overview of the execution of Slide skill. The figure 5.9 shows the ex-

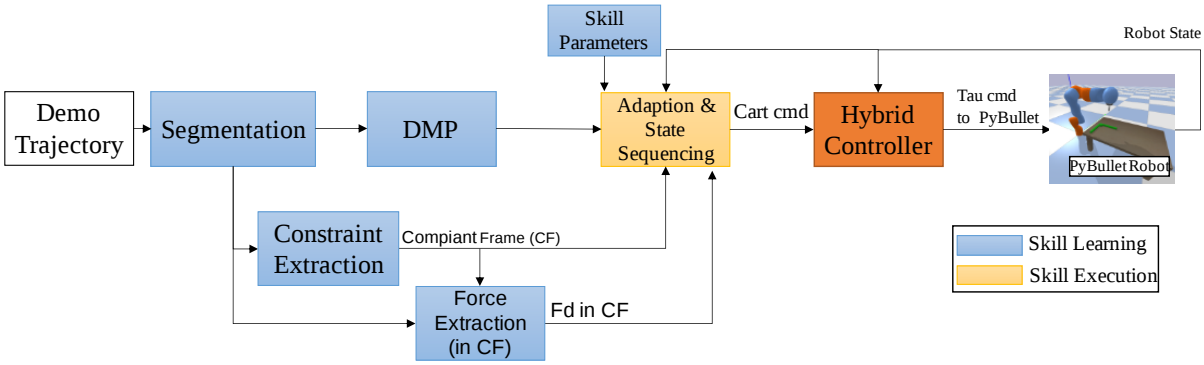
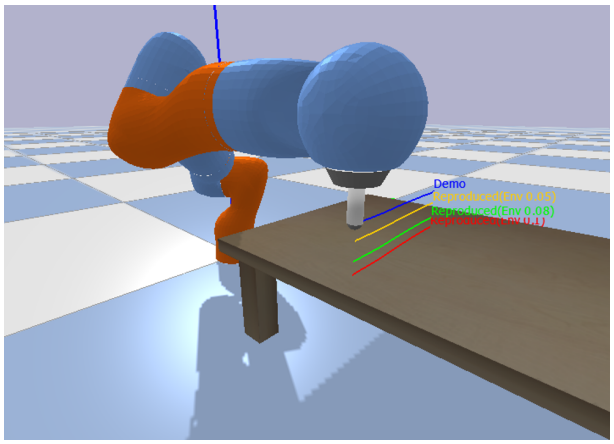


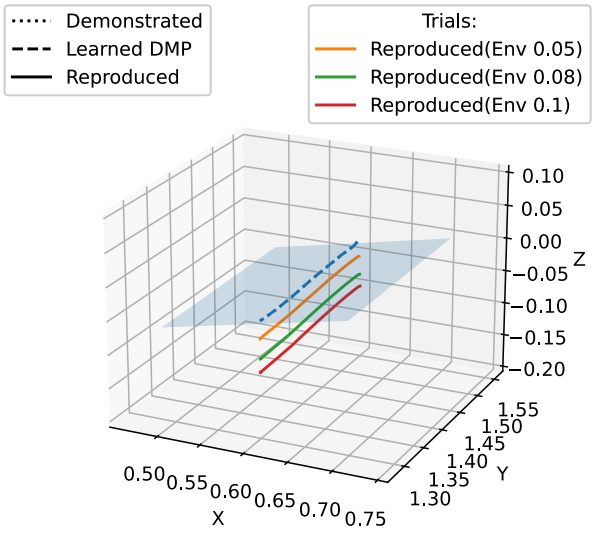

**Figure 5.8.:** Overview of the Execution of Skill 1: Slide

perimental results of reproduction of an example Slide skill which has only one segment. The table in the environment is shifted down about 0.05m, 0.08m and 0.1m respectively during each trial and skill is reproduced without any additional inputs about changes in the environment. Results of each trial are compared with demonstration trajectory and simple DMP learning.

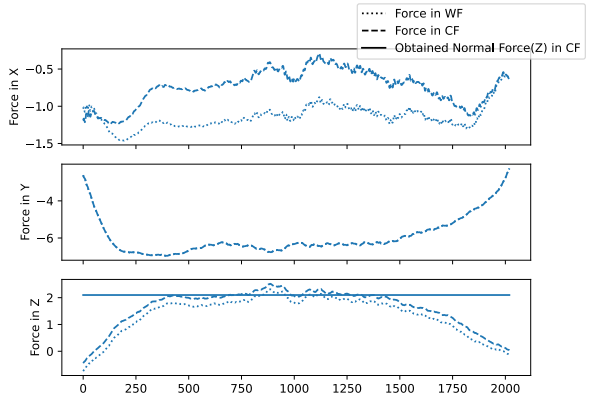
From the Fig. 5.9 it can be interpreted that, with change in the environment, sliding skill adapted the changes and reproduced the skill with same interaction behavior in all the trials. The figure 5.9a shows the comparison of demo and reproduced trajectories of three different trials which is rendered in PyBullet Environment. In the Fig.5.9b, demonstrated trajectory coincides with the trajectory from the learned DMP and reproduced trajectories



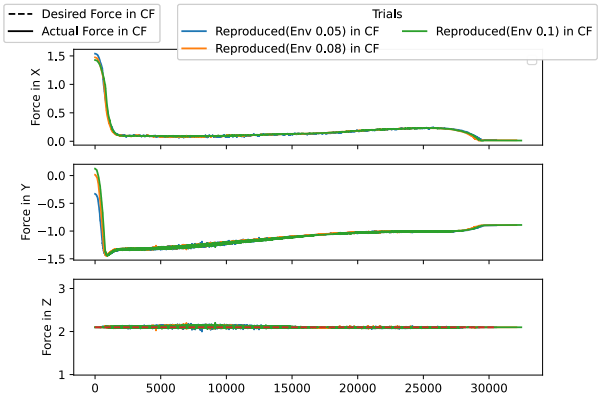

(a) Visualization of 3D Trajectories in PyBullet




(b) 3D Trajectory in World Frame with extracted CF as Plane




(c) Demo Force Profiles in WF and in CF with Learned Normal Force in CF




(d) Reproduced Force Profiles in CF with Learned Normal Force in CF

**Figure 5.9.:** Experimental Results of Skill 1: Slide

exhibit same behavior on the varying environment. The adaption capability of DMPs for novel situations enables the skill to be adapted to new initial and goal points instead of reproducing same trajectory. This behavior can be observed in subsequent skills. For a fair comparison, DMP trajectory with reproduced trajectories without any adaptations to initial and goal state are compared. Extraction of constant normal force in CF is shown in the Fig.5.9c. Constant force is obtained over a low variance region of the Z-axis force profile in CF. Figure 5.9d shows that, slide skill is able to apply a constant normal force in Z direction in Compliant frame for entire trajectory and is same in all trials. Forces in X and Y directions are from the position control in CF and is of no interest to compare, as it is intended to observe the interaction with environment only through normal force in CF.

### 5.2.2. Skill 2: Contour

For a Contour Skill, force application at each point varies and will be learned using DMP. However, instead of controlling forces in all directions, a CF is extracted at each point and forces are applied only in the Z-direction. Position control is enabled in X and Y directions and force control is enabled in Z direction of CF. At first trajectory segmentation is performed using cluster based segmentation then the compliant frame at each point and force that needs to be applied in CF are extracted from the learned force trajectories of the demo data. A smooth trajectory of normalized force from force vector at each point that is applied in the Z direction of CF is obtained. A local coordinate system at each point can be represented as compliant frame, which is extracted by aligning a Z axis of an arbitrary frame with unit direction of force vector at each point. The overview of the Contour Skill execution is shown in Fig.5.10

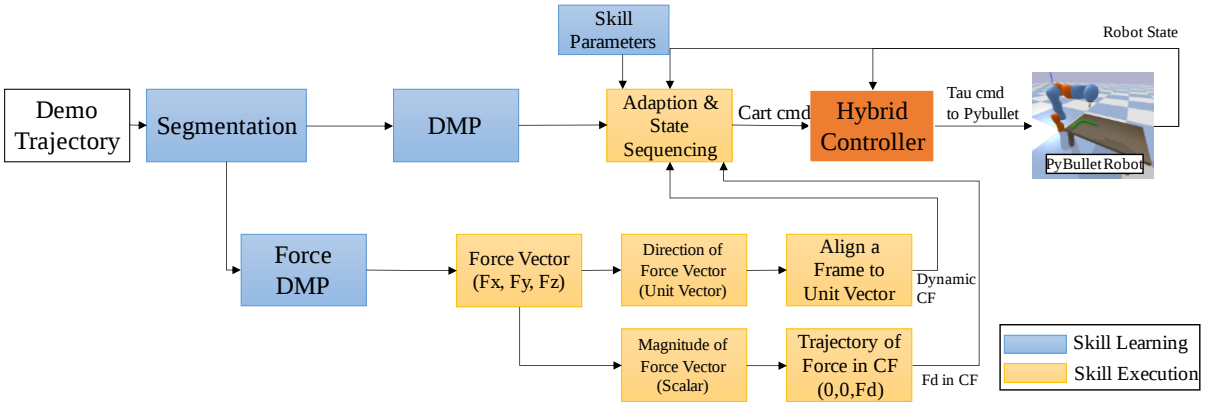
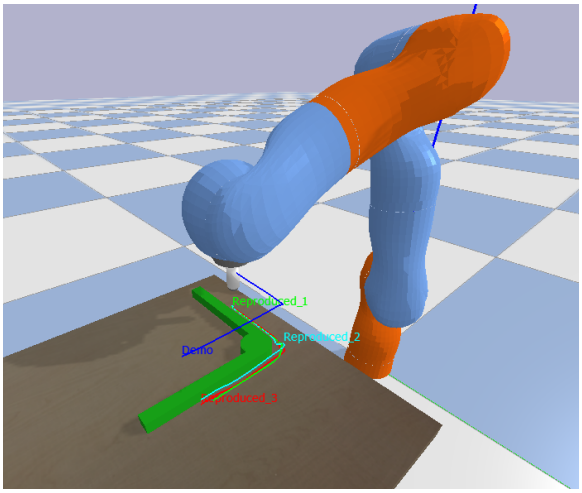


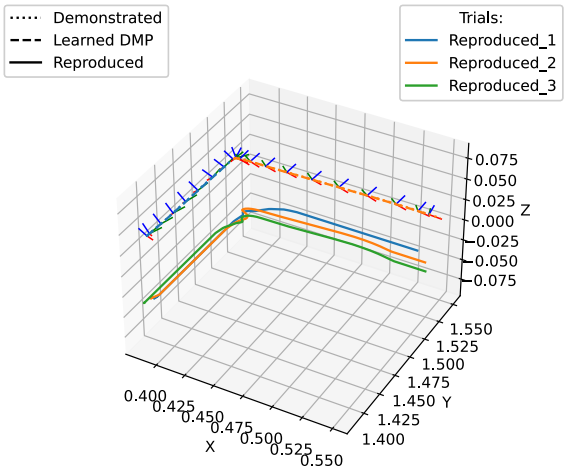

**Figure 5.10.:** Overview of the Execution of Skill 2: Contour

Figure5.11 shows the experimental results of reproduction of Contour skill with varying environment in each trial. Reproduction results are compared with demonstration trajectory and simple DMP learning.

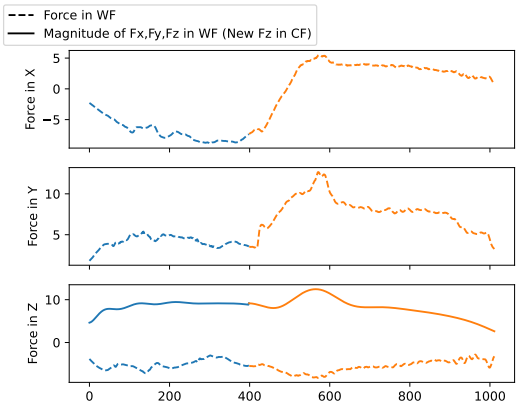
From the Fig. 5.11 it can be interpreted that, with change in the environment, contouring skill adapted the changes and reproduced the skill with same interaction behaviour in all the trials. An example of a contouring task as shown in the Fig.5.11a is considered for conducting the trials. The figure shows the visualization of 3D trajectories in the PyBullet environment. In Fig.5.11b, 3D trajectories of Demo, Learned and Reproduced trials are shown along with the dynamically varying compliant frame that is extracted at each point by aligning the CF Z-axis to the direction of force vector at each point. The demonstrated trajectory represented as dotted lines coincides with the trajectory from the learned DMP represented in dashed lines. The deviation in paths of trials two and three are caused due to more deviation in the environment. A varying smooth Z-Axis force in CF is extracted



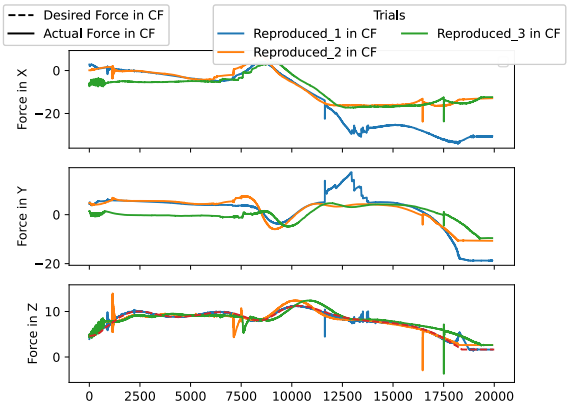

(a) Visualization of 3D Trajectories in PyBullet




(b) 3D Trajectory in World Frame with extracted CF as Plane




(c) Demo Force Profiles in WF and with Learned Force trajectory in Z direction in Varying CF




(d) Reproduced Force Profiles in CF with Learned desired Force in CF

**Figure 5.11.:** Experimental Results of Skill 2: Contour

by computing magnitude of the force vector is shown in the Fig.5.11c. The Figure 5.11d shows that, contour skill is able to interact with the environment by applying varying force in varying Compliant frame at each point. Forces in X and Y are caused due to fixed position control in varying CF. The force profile in all the trials exibit similar behaviour and is able to track the forces in Z direction with desired forces in CF.

### 5.2.3. Skill 3: Touch

For a touch skill, the robot learns to perform touch operation, where forces are not applied on the environment but robot moves until it reaches the environment with learned contact force while demonstration. At first the trajectory is segmented into two as approach and departure paths at maximum force point where robot touches the environment during demonstration. DMPs are learned for two paths and contact force at segmentation point is extracted to track the required touch or contact with the environment as shown in Fig.5.12.

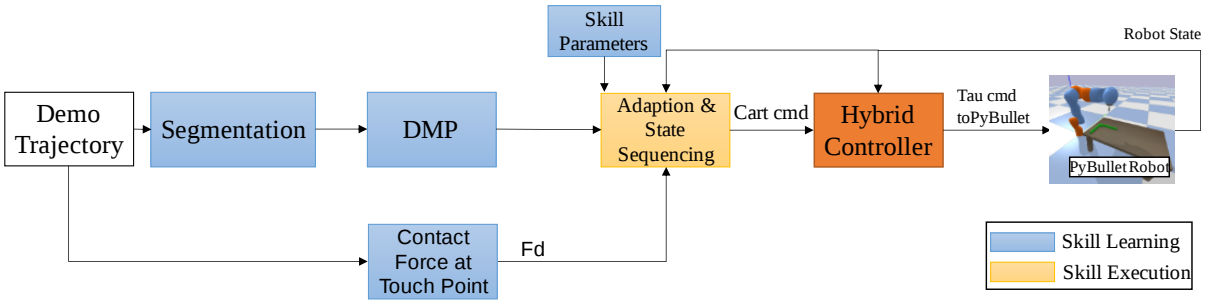



**Figure 5.12.:** Overview of the Execution of Skill 3: Touch

During skill execution, there is a possibility of three different cases as shown in the Fig.4.16. Trajectory points are extracted and in parallel, contact state of the robot is tracked from the feedback of robot state. Until the contact state is occurred, robot continues to move in the path of approach learned from DMP. On the other hand, if contact point has not reached even when approach path learned from DMP is executed, position commands are incremented in small values in the direction of contact force to reach environment. When a contact point is achieved either before intended goal position or after goal position, current position is considered as new initial position for departure path DMP and trajectory commands are generated to move away from the environment in the adapted path for departure. This adaption mechanism and state sequencing during skill execution as depicted in Fig.5.13 ensures that during skill reproduction, robot is able adapt to the environment dynamics.

Figure5.14 shows the experimental results of reproduction of Touch skill with three different interaction scenarios in each trial. Reproduction results are compared with demonstration trajectory and simple DMP learning.

From the Fig. 5.14 it can be interpreted that, with change in the environment, Touch skill adapted the changes and reproduced the skill with same interaction behaviour in all the trials. An example of a Touch task as shown in the Fig.5.14a is considered for conducting the trials. The figure shows the visualization of 3D trajectories in the PyBullet

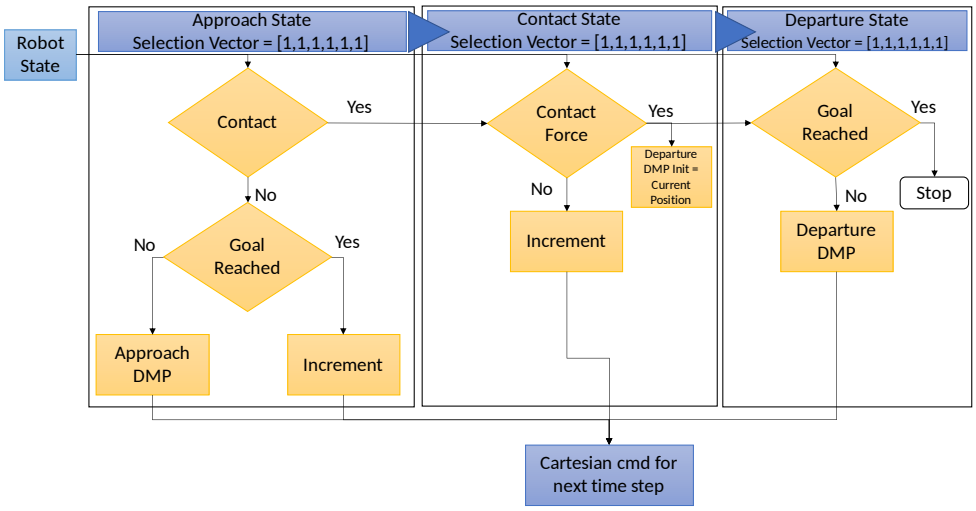
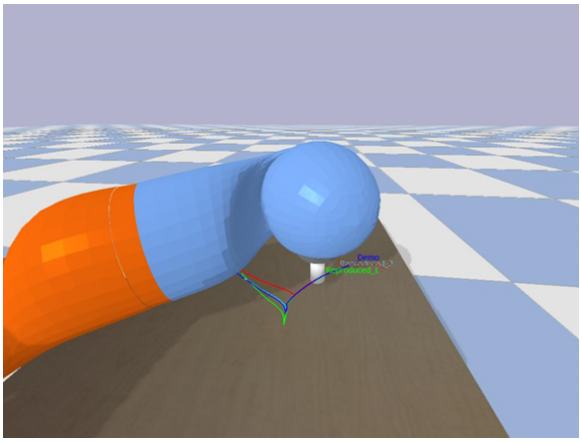


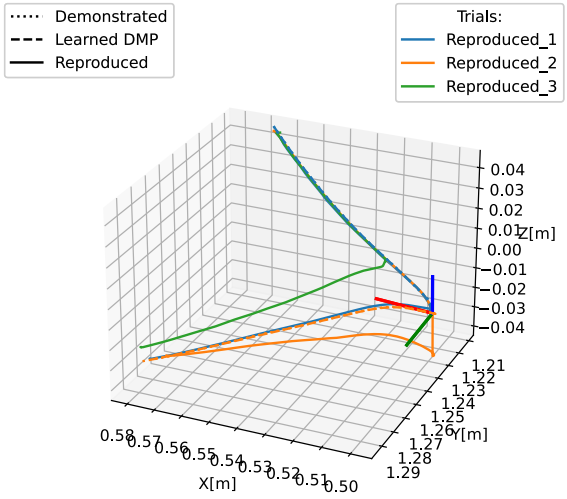

**Figure 5.13.:** State sequencing of Skill Execution for Skill 3: Touch

environment. In Fig.5.14b, 3D trajectories of Demo, Learned and Reproduced trials are shown along with the compliant frame extracted at segmentation point. The demonstrated trajectory represented as dotted lines coincides with the trajectory from the learned DMP represented in dashed lines. Demonstrated forces are used to segment the trajectory into two paths by defining segmentation point at maximum force point or touch point as shown in the Fig.5.14c. The Figure 5.14d shows that, touch skill is able to interact with the environment by identifying the contact point and by applying the defined contact force on to the environment. Contact force at interaction point, can either be applied by extracting from data or can be defined.

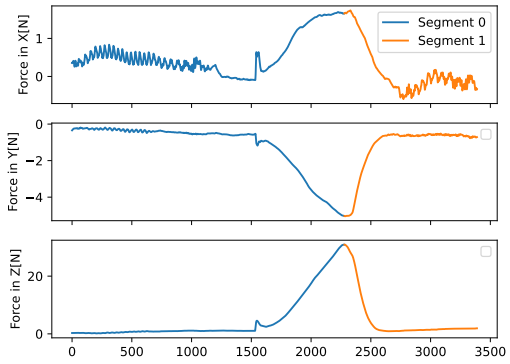
In the trials, a fixed contact force about 1N is considered and such the force profile in Fig.5.14d shows that, when robot reached contact with environment it stays in contact until the defined contact force is applied. After applying the contact force, robot returns in the departure path adapting the contact location as new initial point.



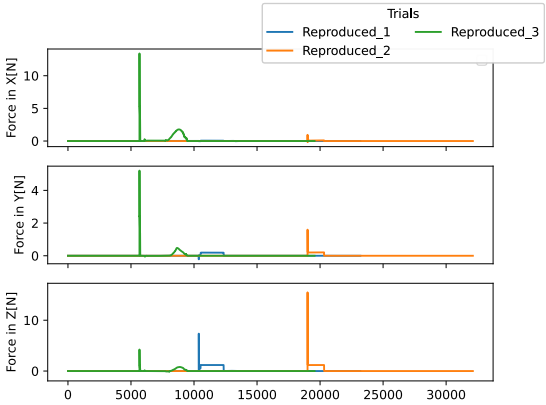
(a) Visualization of 3D Trajectories in PyBullet




(b) 3D Trajectory in World Frame with extracted CF as Plane




(c) Demo Force Profiles in WF with segmentation at max force or touch point




(d) Reproduced Force Profiles in WF

**Figure 5.14.:** Experimental Results of Skill 3: Touch

### 5.2.4. Skill 4: Press

For Press skill, the robot should learn and reproduce the pressing operation, where, robot approaches the environment and applies pressing forces on to the environment. At first during skill learning, demo data is segmented into three segments corresponding to three states as shown in the Fig.4.19. DMPs are learned for position, orientation for approach and departure paths and forces are learned for intermediate segment using DMP. Skill execution is performed by sequencing the states and adapting to environment dynamics based on contact state. The figure 5.15 gives an overview of skill reproduction.

Skill execution is performed in three states, at first, the trajectory points are extracted for each time step and in parallel, contact state of the robot is tracked from the feedback

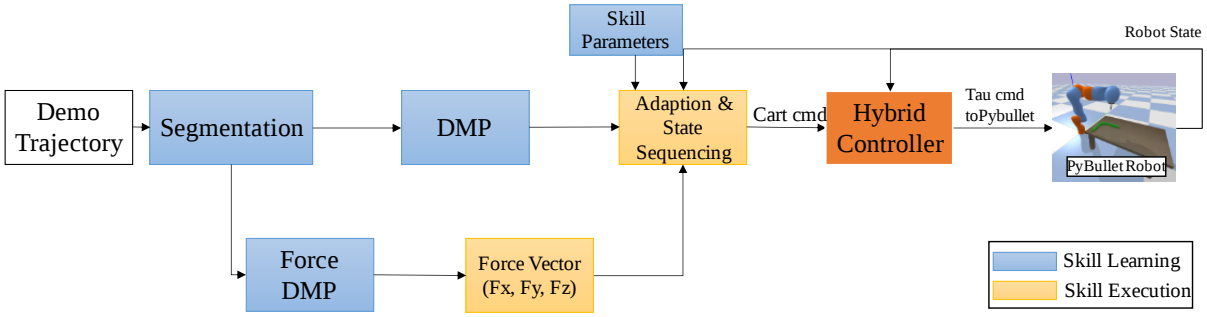



**Figure 5.15.:** Overview of the Execution of Skill 4: Press

of the robot state. The robot moves in the approach path learned using DMP until it reaches the environment. Interaction forces learned from demonstration are applied on the environment. Later, current position is initialized as initial state for departure DMP and robots starts moving away in the adapted departure path. The state sequence and adaption implemented during the skill execution as shown in the Fig.5.16 ensures that during reproduction, robot adapts to the dynamic changes in the environment.

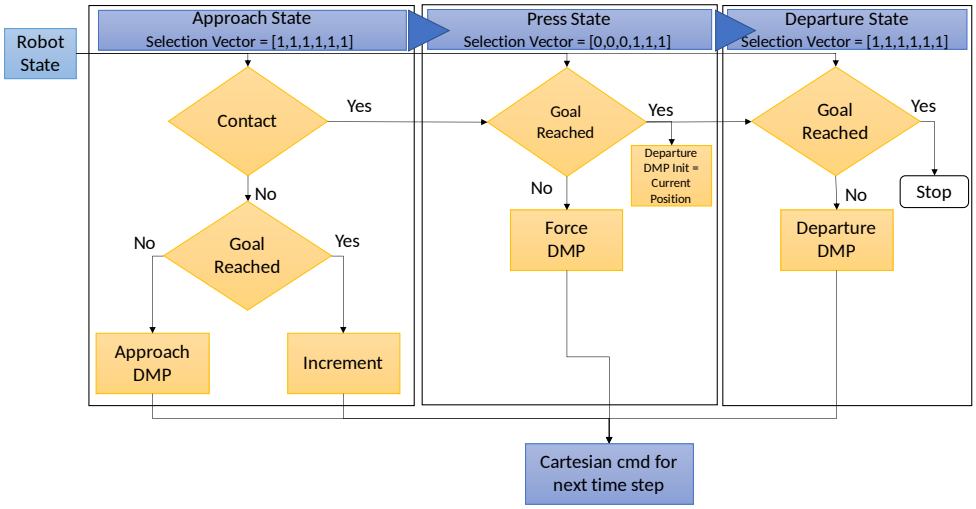
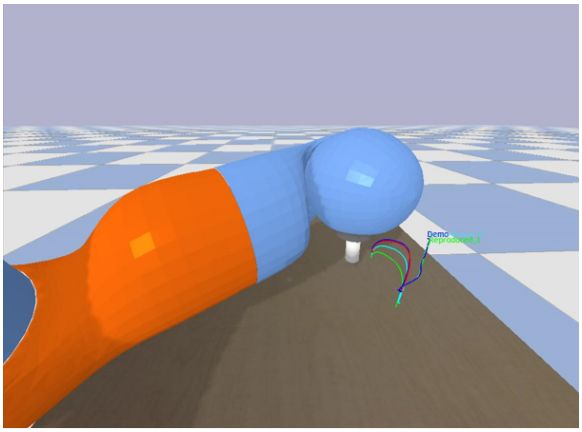


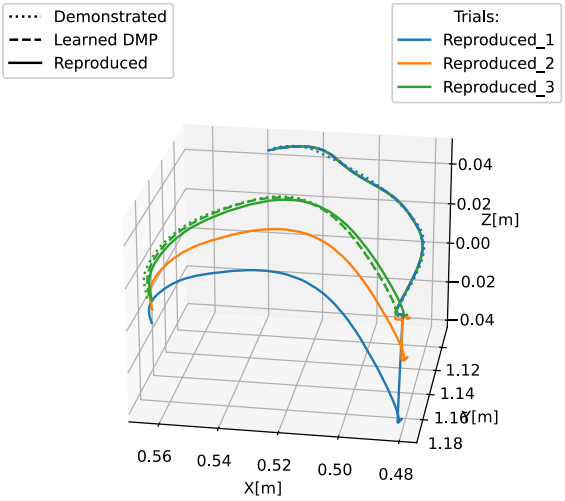

**Figure 5.16.:** State sequencing of Skill Execution for Skill 3: Touch

Figure5.17 shows the experimental results of reproduction of Press skill with three different trials by varying environment in each trial. Reproduction results are compared with demonstration trajectory and simple DMP learning.

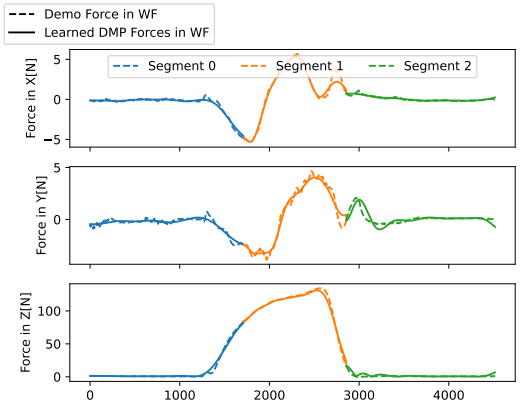
From the Fig. 5.17 it can be interpreted that, with change in the environment, Press skill adapted the changes and reproduced the skill with same interaction behaviour in all the trials. An example of a Press skill as shown in the Fig.5.17a is considered for conducting the trials. The figure shows the visualization of 3D trajectories in the PyBullet environment. In Fig.5.17b, 3D trajectories of Demo, Learned and Reproduced trials are shown. The demonstrated trajectory represented as dotted lines aligns with the trajectory from the learned DMP represented in dashed lines. Demonstrated forces are learned



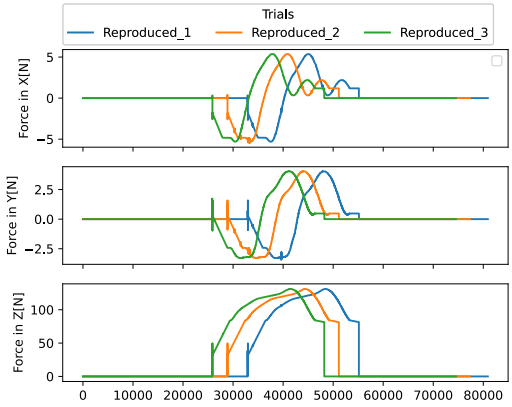
(a) Visualization of 3D Trajectories in PyBullet




(b) 3D Trajectory in World Frame




(c) Demo Force Profiles and with Learned Force trajectory in WF




(d) Reproduced Force Profiles in WF

**Figure 5.17.:** Experimental Results of Skill 3: Touch

using DMPs and are used to apply on the environment during interaction as shown in the Fig.5.17c. The Figure 5.17d shows that, press skill is able to interact with the environment by identifying the contact point and by applying the learned contact forces on to the environment. Though the environment is changed for each trials, applied contact forces at the interaction point on to the environment is similar in all the trials.

## 5.3. Summary

This chapter presented the experimental setup implemented and the evaluation of the approach presented earlier. The results are extracted and evaluated by conducting trials on each skill under simulation environment.

Skills are demonstrated on DLR LWR-IV robot in Zero Gravity Mode and the data is recorded for skill learning. Skill learning based on the approach proposed is to extract the features by learning from demonstrated data. The feature extraction for each skill is implemented in Python module. A skill execution module which is also implemented in Python, at each time step generates a cartesian command from features that are extracted by using skill learning module. A hybrid position-force controller implemented in MATLAB Simulink package receives the cartesian command from skill execution module and generates a torque command to the robot in PyBullet simulation Environment that is configured with DLR LWR-IV robot.

Trials are conducted on four preliminary skills: Slide, Contour, Touch and Press. For each skill, results are interpreted on trials with varying environment. From the results obtained, it can be interpreted that during reproduction each skill adapted to the changes in environment. The results satisfied one of the main objectives to learn and reproduce contact based skills that should be able to interact with the environment with same behavior and adapt to novel situations which is in contrast to simple replay of the trajectories.

# 6. Discussion

This chapter covers the discussion regarding the results and the implications of the approaches proposed for skill based learning framework. The results are analyzed against project goals, requirements and compared with simple trajectory replay. And finally this chapter covers the limitations of the system and potential future work and improvements that could possibly be implemented in the skill based learning framework.

## 6.1. Implications of the Results

By analyzing the results from the previous chapter for four skills, following important points can be inferred:

1. For the skills considered so far, the features identified in chapter 3: Segmentation, Compliant Frame, Selection Matrix, and desired trajectories are enough to represent and reproduce a skill.

2. The hybrid controller architecture devised in chapter 3 satisfied the requirements of features extracted such as rigid transformation of task frames, facilitates intuitive force application using Compliant frames and selection matrix definition in compliant frame. The same controller is utilized for all the skills without any modifications in the strategy. Therefore, the hybrid control strategy employed can be generalized to most of the skills with minimum or no modifications in the strategy for future skills.

3. The idea of identifying a Compliant Frame for each skill and defining selection matrix and stiffness matrix in Compliant Frame allowed the reproduce the contact based skills with similar behavior even when environment is varying.

4. The adaptive capabilities of DMP towards novel situations are well utilized to reproduce the skill with interaction behaviour.

## 6.2. Limitations

The framework presented in this thesis has not considered the application of interactive torque, which limits the use of same features for torque based skills. The skills chosen so far do not require learning of torque profiles. The interactive skills are able to learn only desired contact forces. Therefore, the employed approach has not dealt with learning of torques quantities.

The approximation function used in forcing term of DMPs as in equation 4.3 which is learned using LWR with fixed number of weights $w_i$, which is defined manually. Though the function approximator learned the trajectories well to achieve the goal, there exists approximation errors which increases with complex trajectories. Although, segmentation methods presented reduce the complexity of trajectories, the approximation errors are not considered during segmentation and such errors still exist.

In the approach presented for feature extraction in chapter 4, segmentation method to be employed specific to each skill is provided. Yet, segmentation approaches implemented requires manual intervention to tune parameters based on the task and skills.

## 6.3. Future work

As future work, intended to build a skill library by extending the proposed approach to other skills. The approaches employed to frame skill templates for four skills can be extended further for other commonly used skills in industrial applications like profile skills such as bulb screwing and valve fitting, Peg-In-Hole, collaborative assembly tasks etc. An advanced extension is to integrate skill specific algorithms into the framework for complex skills such as Peg-in-hole tasks.

In addition to the features identified so far, further analysis needs to be carried out as a future work to identify and extract features for complex skills, where torques and orientations are involved in interactive tasks, such as screwing and unscrewing bulb and opening doors. Presented framework in this thesis targeted the interaction with environment with only contact forces. As a future work a learning approach needs to be considered to learn torque profiles either using DMPs or any other alternative methods to generate torque trajectories.

Learning approaches like DMP can be further extended specifically to contact skills that offers better generalization and adaption capability to interact with environment. As proposed in [CH18], an extension for DMP called as contact DMP to adapt to goals

dynamically based on contact state of the robot with environment. Similar approaches can be employed to deal with contact based skills. Alternatively, in order to deal with such contact based skills, MPs can be further extended to have closed loop architecture to track the state and generate trajectories online while executing the task instead of generating trajectories beforehand.

As explained earlier, the approximation errors due to the function approximation using LWR may not represent a good imitation behavior of the demonstrated task. Therefore, better approximation methods need to be employed to estimate the parameters accordingly that are defined manually in current LWR method such as weights. The Receptive Field Weighted Regression (RFWR) is a suitable alternative for function approximation proposed in [Sch99]. Alternatively, more sophisticated methods based on task parameterization methods such as using Gaussian Process(GP) or Gaussian Mixture Regression (GMR) as proposed in [Cal16] [PL18] can be explored further for possible extension of DMPs and utilize for contact based tasks.

More sophisticated segmentation algorithms, which facilitate minimal user inputs will be an add-on for building skill library. Also, though the segmentation simplifies learning of complex trajectories by representing each segment as Movement Primitive, there is a possibility of over segmentation or under segmentation for potential Movement Primitive representations. A methodology to combine multiple segments as MPs such as DND algorithm as mentioned in [MGK16] needs to be employed for developing skill templates.

Due to non-availability of robotic system, proposed approach for skill based learning is entirely evaluated under simulation environment. Although, the demonstrated data is obtained from physical robot, the skills are learned and reproduced on a simulated robot under PyBullet simulation environment. Though the results obtained are reliable and satisfactory, practical issues such as varying friction, dynamical uncertainties in the environment and disturbances in mechanical, electronics and computing modules are ignored. For better evaluation, it is recommended to perform trials on physical robot. As a future work the experiments should be conducted on multiple robotic systems and results should be compared to gain more practical insights and better understanding of the skill based learning framework.

# 7. Conclusion

The main objective of this thesis work is to formulate a methodology to develop skill based learning framework for contact based tasks. Approaches that are parameterized more specific to each skill are employed in contrast to the generic approach by identifying and extracting important features that represent a skill template. These features are used to learn and reproduce the skill that displayed generalization and adaptive capabilities with dynamically changing environments.

The skill specific approach for learning, which only requires a single demonstration of the task is clearly a benefit over other LfD techniques that require more than one demonstration. Segmentation of movements and skill specific learning of features that are engineered specifically for skills allowed the learning of complex tasks with a single demonstration. The adaption and generalization capabilities of Dynamic Movement Primitives are well utilized to adapt to the contact states with environment during skill reproduction.

To achieve the objective of reproducing contact based skills, a control strategy: A Hybrid Position-Force controller is proposed for simultaneous control of position, orientation and force quantities that also offers rigid transformations of compliant frames to reproduce interaction skills.

Proposed methodology for skill parameterization was implemented over multiple modules, which were interfaced through a realtime communication middle-ware. The implementation was evaluated under PyBullet Simulation Environment configured with similar robot model DLR LWR-IV that was used for demonstration. The results were extracted and interpreted by reproducing four preliminary skills: Slide, Contour, Touch and Press with varying environments. From the results, it can be inferred that by using the methodology that was proposed and implemented, skills were reproduced as per desired behavior by learning from single demonstration and adapted to the changes in environment. The results satisfied the main objectives to learn and reproduce contact based skills that should be able to interact with the environment with same behavior and adapted to novel situations which is in contrast to simple replay of the trajectories.

Limitations of the presented methodology was analyzed and future works are proposed for further development of skill templates for other important skills. Embedding skill specific algorithms and establishing contact or closed loop DMPs adapted for contact based tasks are some of the potential future works relevant to contact based skills.

## Bibliography


[ABM90] Abbati-Marescotti, A.; Bonivento, C.; Melchiorri, C.
*On the invariance of the hybrid position/force control*
In: Journal of Intelligent and Robotic Systems, 3 (1990) 3, DOI 10.1007/BF00126071, pp. 233–250.

[AHO07] Albu-Schäffer, A.; Haddadin, S.; Ott, C.; Stemmer, A.; Wimböck, T.; Hirzinger, G.
*The DLR lightweight robot: Design and control concepts for robots in human environments*
In: Industrial Robot, 34 (2007) 5, DOI 10.1108/01439910710774386, pp. 376–385.

[AMS97] Atkeson, C. G.; Moore, A. W.; Schaal, S.
*Locally Weighted Learning*
In: Artificial Intelligence Review, 11 (1997) 1-5, DOI 10.1007/978-94-017-2053-3_2, pp. 11–73.

[BBK96] Bakker, P.; Bakker, P.; Kuniyoshi, Y.
*Robot See, Robot Do : An Overview of Robot Imitation*
In: IN AISB96 WORKSHOP ON LEARNING IN ROBOTS AND ANIMALS (, 1996), http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.27.292, pp. 3–11.

[BD96] Bruyninckx, H.; De Schutter, J.
*Specification of force-controlled actions in the "Task frame formalism" - A synthesis*
In: IEEE Transactions on Robotics and Automation, 12 (1996) 4, DOI 10.1109/70.508440, pp. 581–589.

[BG13] Billard, A.; Grollman, D.
*Robot learning by demonstration*
In: Scholarpedia, 8 (2013) 12, DOI 10.4249/scholarpedia.3824, http://www.scholarpedia.org/article/Robot%7B%5C_%7Dlearning%7B%5C_%7Dby%7B%5C_%7Ddemonstration, p. 3824.

[BNP12] Bøgh, S.; Nielsen, O. S.; Pedersen, M. R.; Krüger, V.; Madsen, O.
*Does your Robot have Skills?*
In: Proceedings of the 43rd International Symposium on Robotics (, 2012),

http://forskningsbasen.deff.dk/Share.external?sp=S9f184d42-459a-4d30-871c-bab8082cfbba%7B%5C&%7Dsp=Saau.

[Cal16] Calinon, S.
*A tutorial on task-parameterized movement learning and retrieval*
In: Intelligent Service Robotics, 9 (2016) 1, DOI 10.1007/s11370-015-0187-9, pp. 1–29.

[CFS10] Calinon, S.; Florent, D.; Sauser, E. L.; Caldwell, D. G.; Billard, A. G.
*An approach based on Hidden Markov Model and Gaussian Mixture Regression*
In: IEEE Robotics and Automation Magazine, 17 (2010) June, pp. 44–45.

[CGB07] Calinon, S.; Guenter, F.; Billard, A.
*On learning, representing, and generalizing a task in a humanoid robot*
In: IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 37 (2007) 2, DOI 10.1109/TSMCB.2006.886952, pp. 286–298.

[CH18] Conkey, A.; Hermans, T.
*Learning Task Constraints from Demonstration for Hybrid Force/Position Control*
In: arXiv (, 2018), pp. 162–169.

[CL18] Calinon, S.; Lee, D.
*Learning Control*
In: Humanoid Robotics: A Reference (, 2018), DOI 10.1007/978-94-007-7194-9_68-2, pp. 1–52.

[CNS99] Caccavale, F.; Natale, C.; Siciliano, B.; Villani, L.
*Six-DOF impedance control based on angle/axis representations*
In: IEEE Transactions on Robotics and Automation, 15 (1999) 2, DOI 10.1109/70.760350, pp. 289–300.

[CR79] Craig, J. J.; Raibert, M. H.
*A systematic method of hybrid position/force control of a manipulator*
In: pp. 446–451.

[CT12] Cakmak, M.; Thomaz, A. L.
*Designing robot learners that ask good questions*
In: HRI'12 - Proceedings of the 7th Annual ACM/IEEE International Conference on Human-Robot Interaction (, 2012), DOI 10.1145/2157689.2157693, pp. 17–24.

[DSD01] Duindam, V.; Stramigioli, S.; Duindam, V.
*Variable Spatial Springs for Robot Control Applications*
In: IEEE International Conference on Intelligent Robots and Systems, 4 (2001), DOI 10.1109/iros.2001.976352, pp. 1906–1911.

[E C16] E. Coumans and Y. Bai
*Pybullet, a Python Module for Physics Simulation for Games, Robotics and Machine Learning*
https://pybullet.org.

[EGC09] Evrard, P.; Gribovskaya, E.; Calinon, S.; Billard, A.; Kheddar, A.
*Teaching physical collaborative tasks: Object-lifting case study with a humanoid*
In: 9th IEEE-RAS International Conference on Humanoid Robots, HUMANOIDS09 (, 2009), DOI 10.1109/ICHR.2009.5379513, pp. 399–404.

[ESL19] Eiband, T.; Saveriano, M.; Lee, D.
*Learning haptic exploration schemes for adaptive task execution*
In: Proceedings - IEEE International Conference on Robotics and Automation, 2019-May (2019), DOI 10.1109/ICRA.2019.8793934, pp. 7048–7054.

[FB97] Fasse, E. D.; Broenink, J. F.
*A Spatial Impedance Controller for Robotic Manipulation*
In: IEEE Transactions on Robotics, 13 (1997) 4, pp. 546–556.

[GLX19] Gao, X.; Ling, J.; Xiao, X.; Li, M.
*Learning Force-Relevant Skills from Human Demonstration*
In: Complexity, 2019 (2019), DOI 10.1155/2019/5262859.

[HAK03] Hwang, J. H.; Arkin, R. C.; Kwon, D. S.
*Mobile robots at your fingertip: Bezier curve on-line trajectory generation for supervisory control*
In: IEEE International Conference on Intelligent Robots and Systems, 2 (2003) October, DOI 10.1109/iros.2003.1248847, pp. 1444–1449.

[HGC08] Hersch, M.; Guenter, F.; Calinon, S.; Billard, A.
*Dynamical system modulation for robot learning via kinesthetic demonstrations*
In: IEEE Transactions on Robotics, 24 (2008) 6, DOI 10.1109/TRO.2008.2006703, pp. 1463–1467.

[Hog84] Hogan, N.
*IMPEDANCE CONTROL: AN APPROACH TO MANIPULATION.*
In: pp. 304–313.

[INH13] Ijspeert, A. J.; Nakanishi, J.; Hoffmann, H.; Pastor, P.; Schaal, S.
*Dynamical movement primitives: Learning attractor models formotor behaviors*
In: Neural Computation, 25 (2013) 2, DOI 10.1162/NECO_a_00393, pp. 328–373.

[JWH13] James, G.; Witten, D.; Hastie, T.; Tibshirani, R.
*Springer Texts in Statistics An Introduction to Statistical Learning - with Applications in R*
2013, ISBN 9781461471370.

[KB10] Khansari-Zadeh, S. M.; Billard, A.
*Imitation learning of globally stable non-linear point-to-point robot motions using nonlinear programming*
In: IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings (, 2010), DOI 10.1109/IROS.2010.5651259, pp. 2676–2683.

[KB11] Khansari-zadeh, S. M.; Billard, A.
*Learning Stable Nonlinear Dynamical Systems With Gaussian Mixture Models*
In: Transactions on R, 27 (2011) 5, pp. 943–957.

[KCC11] Kormushev, P.; Calinon, S.; Caldwell, D. G.
*Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input*
In: Advanced Robotics, 25 (2011) 5, DOI 10.1163/016918611X558261, pp. 581–603.

[KGN17] Kramberger, A.; Gams, A.; Nemec, B.; Chrysostomou, D.; Madsen, O.; Ude, A.
*Generalization of orientation trajectories and force-torque profiles for robotic assembly*
In: Robotics and Autonomous Systems, 98 (2017), DOI 10.1016/j.robot.2017.09.019, pp. 333–346.

[KGS15] Kober, J.; Gienger, M.; Steil, J. J.
*Learning movement primitives for force interaction tasks*
In: Proceedings - IEEE International Conference on Robotics and Automation, 2015-June (2015) June, DOI 10.1109/ICRA.2015.7139639, pp. 3192–3199.

[Kha87] Khatib, O.
*A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation*
In: IEEE Journal on Robotics and Automation, 3 (1987) 1, DOI 10.1109/JRA.1987.1087068, pp. 43–53.

[KTN08] Kulić, D.; Takano, W.; Nakamura, Y.
*Combining automated on-line segmentation and incremental clustering for whole body motions*
In: Proceedings - IEEE International Conference on Robotics and Automation (, 2008), DOI 10.1109/ROBOT.2008.4543603, pp. 2591–2598.

[LD88] Lipkin, H.; Duffy, J.
*Hybrid twist and wrench control for a robotic manipulator*
pp. 138–144, DOI 10.1115/1.3258918.

[Lee17] Lee, J.
*A survey of robot learning from demonstrations for Human-Robot Collaboration*
In: arXiv (, 2017), arXiv: 1710.08789.

[LGM12] Luksch, T.; Gienger, M.; Muhlig, M.; Yoshiike, T.
*Adaptive movement sequences and predictive decisions based on hierarchical dynamical systems*
In: IEEE International Conference on Intelligent Robots and Systems (, 2012), DOI 10.1109/IROS.2012.6385651, pp. 2082–2088.

[LKK16] Lin, J. F. S.; Karg, M.; Kulić, D.
*Movement Primitive Segmentation for Human Motion Modeling: A Framework for Analysis*
In: IEEE Transactions on Human-Machine Systems, 46 (2016) 3, DOI 10.1109/THMS.2015.2493536, pp. 325–339.

[LKY13] Lucia, P.; Keisuke, U.; Yoshihiko, N.; Aude, B.
*Learning Robot Skills Through Motion Segmentation and Constraints Extraction*
In: Proceedings of the Collaborative Manipulation Workshop at the ACM/IEEE International Conference on Human-Robot Interaction (HRI 2013) (, 2013).

[LO11] Lee, D.; Ott, C.
*Incremental kinesthetic teaching of motion primitives using the motion refinement tube*
In: Autonomous Robots, 31 (2011) 2-3, DOI 10.1007/s10514-011-9234-3, pp. 115–131.

[Lon87] Lončarić, J.
*Normal Forms of Stiffness and Compliance Matrices*
In: IEEE Journal on Robotics and Automation, 3 (1987) 6, DOI 10.1109/JRA.1987.1087148, pp. 567–572.

[LRS15] Lemme, A.; Reinhart, R. F.; Steil, J. J.
*Self-supervised bootstrapping of a movement primitive library from complex trajectories*
In: IEEE-RAS International Conference on Humanoid Robots, 2015-Febru (2015), DOI 10.1109/HUMANOIDS.2014.7041443, pp. 726–732.

[LT06] Lavielle, M.; Teyssière, G.
*Detection of multiple change-points in multivariate time series*
In: Lithuanian Mathematical Journal, 46 (2006) 3, DOI 10.1007/s10986-006-0028-9, pp. 287–306.

[MGH09] Muhlig, M.; Gienger, M.; Hellbach, S.; Steil, J. J.; Goerick, C.
*Task-level imitation learning using variance-based movement optimization*
In: ( Aug. 2009), DOI 10.1109/robot.2009.5152439, pp. 1177–1184.

[MGK16] Manschitz, S.; Gienger, M.; Kober, J.; Peters, J.
*Probabilistic decomposition of sequential force interaction tasks into movement primitives*
In: IEEE International Conference on Intelligent Robots and Systems, 2016-Novem (2016), DOI 10.1109/IROS.2016.7759577, pp. 3920–3927.

[MGK20] Manschitz, S.; Gienger, M.; Kober, J.; Peters, J.
*Learning sequential force interaction skills*
In: Robotics, 9 (2020) 2, DOI 10.3390/ROBOTICS9020045.

[MLS17] Murray, R. M.; Li, Z.; Shankar Sastry, S.
*A mathematical introduction to robotic manipulation*
2017, ISBN 9781351469791, DOI 10.1201/9781315136370.

[MW16] Marin, A. G.; Weitschat, R.
*Unified impedance and hybrid force-position controller with kinestatic filtering*
In: IEEE International Conference on Intelligent Robots and Systems, 2016-Novem (2016), DOI 10.1109/IROS.2016.7759516, pp. 3353–3359.

[NOA] Niekum, S.; Osentoski, S.; Atkeson, C.; Barto, A.
*Learning Articulation Changepoint Models from Demonstration*
In: Cs.Cornell.Edu ().

[NOA15] Niekum, S.; Osentoski, S.; Atkeson, C. G.; Barto, A. G.
*Online Bayesian changepoint detection for articulated motion models*
In: Proceedings - IEEE International Conference on Robotics and Automation, 2015-June (2015) June, DOI 10.1109/ICRA.2015.7139383, pp. 1468–1475.

[NP08] Nguyen-Tuong, D.; Peters, J.
*Local Gaussian process regression for real-time model-based robot control*
In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (, 2008), DOI 10.1109/IROS.2008.4650850, pp. 380–385.

[NS15] Neumann, K.; Steil, J. J.
*Learning robot motions with stable dynamical systems under diffeomorphic transformations*
In: Robotics and Autonomous Systems, 70 (2015), DOI 10.1016/j.robot.2015.04.006, http://dx.doi.org/10.1016/j.robot.2015.04.006, pp. 1–15.

[OSR14] (OSRF), O. S. R. F.
*The Robot Operating System (ROS) Website*
In: ( 2014), http://www.ros.org/.

[Par17] Paraschos, A.
*Robot Skill Representation, Learning and Control with Probabilistic Movement Primitives*
In: PhD - TU Darmstadt (, 2017), http://tuprints.ulb.tu-darmstadt.de/id/eprint/6947%7B%5C%%7D0Ahttp://tuprints.ulb.tu-darmstadt.de.

[PKM13] Pastor, P.; Kalakrishnan, M.; Meier, F.; Stulp, F.; Buchli, J.; Theodorou, E.; Schaal, S.
*From dynamic movement primitives to associative skill memories*
In: Robotics and Autonomous Systems, 61 (2013) 4, DOI 10.1016/j.robot.2012.09.017, http://dx.doi.org/10.1016/j.robot.2012.09.017, pp. 351–361.

[PL18] Pervez, A.; Lee, D.
*Learning task-parameterized dynamic movement primitives using mixture of GMMs*
In: Intelligent Service Robotics, 11 (2018) 1, DOI 10.1007/s11370-017-0235-8, pp. 61–78.

[PMS13] Park, S.; Mustafa, S. K.; Shimada, K.
*Learning-Based Robot Control with Localized Sparse Online Gaussian Process*
In: IEEE International Conference on Intelligent Robots and Systems (, 2013), DOI 10.1109/IROS.2013.6696503, pp. 1202–1207.

[PNA16] Pedersen, M. R.; Nalpantidis, L.; Andersen, R. S.; Schou, C.; Bøgh, S.; Krüger, V.; Madsen, O.
*Robot skills for manufacturing: From concept to industrial deployment*
In: Robotics and Computer-Integrated Manufacturing, 37 (Feb. 2016), DOI 10.1016/j.rcim.2015.04.002, pp. 282–291.

[PS16] Perrin, N.; Schlehuber-Caissier, P.
*Fast diffeomorphic matching to learn globally asymptotically stable nonlinear dynamical systems*
In: Systems and Control Letters, 96 (Oct. 2016), DOI 10.1016/j.sysconle.2016.06.018, pp. 51–59.

[RCC16] Rozo, L.; Calinon, S.; Caldwell, D. G.; Jiménez, P.; Torras, C.
*Learning Physical Collaborative Robot Behaviors From Human Demonstrations*
In: IEEE Transactions on Robotics, 32 (2016) 3, DOI 10.1109/TRO.2016.2540623, pp. 513–527.

[Roc13] Rock
*ROCK, the Robot Construction Kit*
http://www.rock-robotics.org.

[RW] Rasmussen, C. E.; Williams, C. K. I.
*Gaussian Processes for Machine Learning*
www.GaussianProcess.org/gpml.

[Sch99] Schaal, S.
*Is imitation learning the route to humanoid robots?*
pp. 233–242, DOI 10.1016/S1364-6613(99)01327-3, https://pubmed.ncbi.nlm.nih.gov/10354577/.

[SL13] Soltoggio, A.; Lemme, A.
*Movement primitives as a robotic tool to interpret trajectories through learning-by-doing*
In: International Journal of Automation and Computing, 10 (2013) 5, DOI 10.1007/s11633-013-0734-9, pp. 375–386.

[SLG18] Shao, Z.; Li, Y.; Guo, Y.; Zhou, X.
*Describing local reference frames for 3-d motion trajectory recognition*
In: IEEE Access, 6 (2018), DOI 10.1109/ACCESS.2018.2849690, pp. 36115–36121.

[SMK15] Steinmetz, F.; Montebelli, A.; Kyrki, V.
*Simultaneous kinesthetic teaching of positional and force requirements for sequential in-contact tasks*
In: IEEE-RAS International Conference on Humanoid Robots, 2015-Decem (2015), DOI 10.1109/HUMANOIDS.2015.7363552, pp. 202–209.

[SPN03] Schaal, S.; Peters, J.; Nakanishi, J.
*Control, planning, learning, and imitation with dynamic movement primitives*
In: Neuroscience (, 2003), pp. 1–21.

[SPN05] Schaal, S.; Peters, J.; Nakanishi, J.; Ijspeert, A.
*Learning movement primitives*
In: Springer Tracts in Advanced Robotics, 15 (2005) January, DOI 10.1007/11008941_60, https://link.springer.com/chapter/10.1007/11008941%7B%5C_%7D60, pp. 561–572.

[ST16] Stenmark, M.; Topp, E. A.
*From demonstrations to skills for high-level programming of industrial robots*
In: AAAI Fall Symposium - Technical Report, FS-16-01 - (2016), pp. 75–78.

[SZG18] Subramani, G.; Zinn, M.; Gleicher, M.
*Inferring geometric constraints in human demonstrations*
In: arXiv (, 2018) CoRL, arXiv: 1810.00140, pp. 1–14.

[TJM99] Tomas, M.; John F, H.; Möller, T.; Hughes, J. F.
*Efficitently Building a Matrix to Rotate One Vector to Another*
In: Journal of Graphics Tools, 4 (Jan. 1999) 4, DOI 10.1080/10867651.1999.10487509, http://www.tandfonline.com/doi/abs/10.1080/10867651.1999.10487509%20http://library1.nida.ac.th/termpaper6/sd/2554/19755.pdf, pp. 7–9.

[TOK07] Takamatsu, J.; Ogawara, K.; Kimura, H.; Ikeuchi, K.
*Recognizing Assembly Tasks Through Human Demonstration*
In: The International Journal of Robotics Research, 26 (July 2007) 7, DOI 10.1177/0278364907080736, http://journals.sagepub.com/doi/10.1177/0278364907080736, pp. 641–659.

[Ude93] Ude, A.
*Trajectory generation from noisy positions of object features for teaching robot paths*
In: Robotics and Autonomous Systems, 11 (Sept. 1993) 2, DOI 10.1016/0921-8890(93)90015-5, pp. 113–127.

[UNP14] Ude, A.; Nemec, B.; Petrič, T.; Morimoto, J.
*Orientation in Cartesian space dynamic movement primitives*
In: Proceedings - IEEE International Conference on Robotics and Automation (, 2014) 4, DOI 10.1109/ICRA.2014.6907291, pp. 2997–3004.

[UUN15] Ureche, A. L. P.; Umezawa, K.; Nakamura, Y.; Billard, A.
*Task Parameterization Using Continuous Constraints Extracted from Human Demonstrations*
In: IEEE Transactions on Robotics, 31 (2015) 6, DOI 10.1109/TRO.2015.2495003, pp. 1458–1471.

[WL18] Wilfinger, L. S.; Lee S, W.
*A Comparison of Force Control Algorithms for Robots in Contact with Flexible Environments*
In: Journal of Materials Processing Technology, 1 (2018) 1, arXiv: arXiv:1011.1669v3, http://dx.doi.org/10.1016/j.cirp.2016.06.001%7B%5C%%7D0Ahttp://dx.doi.org/10.1016/j.powtec.2016.12.055%7B%5C%%7D0Ahttps://doi.org/10.1016/j.ijfatigue.2019.02.006%7B%5C%%7D0Ahttps://doi.org/10.1016/j.matlet.2019.04.024%7B%5C%%7D0Ahttps://doi.org/10.1016/j.matlet.2019.127252%7B%5C%%7D0Ahttp://dx.doi.o, pp. 1–8.

[XHL19] Xu, J.; Hou, Z.; Liu, Z.; Qiao, H.
*Compare contact model-based control and contact model-free learning: A survey of robotic peg-in-hole assembly strategies*
In: arXiv (, 2019) March, arXiv: 1904.05240, pp. 1–15.

[XLE18] Xie, X.; Liu, H.; Edmonds, M.; Gaol, F.; Qi, S.; Zhu, Y.; Rothrock, B.; Zhu, S. C.
*Unsupervised learning of hierarchical models for hand-object interactions*
In: Proceedings - IEEE International Conference on Robotics and Automation (, 2018), DOI 10.1109/ICRA.2018.8461214, pp. 4097–4102.

[XSX16] Xiong, C.; Shukla, N.; Xiong, W.; Zhu, S. C.
*Robot learning with a spatial, temporal, and causal and-or graph*
In: Proceedings - IEEE International Conference on Robotics and Automation, 2016-June (2016), DOI 10.1109/ICRA.2016.7487364, pp. 2144–2151.

[ZCC16] Zeestraten, M. J.; Calinon, S.; Caldwell, D. G.
*Variable duration movement encoding with minimal intervention control*
In: Proceedings - IEEE International Conference on Robotics and Automation, 2016-June (2016), DOI 10.1109/ICRA.2016.7487171, pp. 497–503.

[ZH18] Zhu, Z.; Hu, H.
*Robot learning from demonstration in robotic assembly: A survey*
In: Robotics, 7 (2018) 2, DOI 10.3390/robotics7020017.

[ZMT13] Zhou, F.; Member, S.; Torre, F. D.; Hodgins, J. K.
*for Temporal Clustering of Human Motion*
In: Ieee Transactions on Pattern Analysis and Machine Intelligence, 35 (2013) X, pp. 1–15.

## List of Tables

**List of Figures**

# A. Appendix

## A.1. Movement Primitives

### A.1.1. Locally weighted Regression

Linear regression aims to find a relationship between input and outputs such that $Y = XA$, where, $Y$ is output and $X$ is input data and aim to find solution by minimizing L2 norm or least square error.

$$\hat{\boldsymbol{A}} = \arg\min_{\boldsymbol{A}} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{A}\|^2 \tag{A.1}$$

By differentiating above equation with respect to $A$ and equating it to zero, we obtain solution for the $A$ for minimum least squares error.

$$\hat{\boldsymbol{A}} = X^T(XX^T)^{-1}Y \tag{A.2}$$

Extending the above simple regression with weighted functions using $\phi(X_n)$ RBF defined below, helps to fit nonlinear function over data to obtain relationship between input $\boldsymbol{X} \in \mathbb{R}^{n\times n}$ and output $\boldsymbol{Y} \in \mathbb{R}^{n\times n}$.

$$\tilde{\phi}_k(\boldsymbol{x}_n) = \exp\left(-\frac{1}{2}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)\right) \tag{A.3}$$

where $\mu_k$ and $\Sigma_k$ are the parameters of the $k^{th}$ RBF. An associated diagonal matrix can be written as:

$$\boldsymbol{W}_k = \mathrm{diag}\left(\phi_k(\boldsymbol{x}_1), \phi_k(\boldsymbol{x}_2), \ldots, \phi_k(\boldsymbol{x}_N)\right) \tag{A.4}$$

Similar to equations A.1.1 we obtain solution by minimizing the weighted L2 norm:

$$\hat{\boldsymbol{A}} = \arg\min_{\boldsymbol{A}} (\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{A})^\top \boldsymbol{W}(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{A}) \tag{A.5}$$

We obtain a solution for $A$ as follows:

$$\hat{\boldsymbol{A}} = \boldsymbol{W}\boldsymbol{X}^\top \left(\boldsymbol{X}\boldsymbol{W}\boldsymbol{X}^\top\right)^{-1}\boldsymbol{Y} \tag{A.6}$$

and finally we extract results from the regressor as:

$$\boldsymbol{Y} = \sum_{k=1}^{K} \boldsymbol{W}_k \boldsymbol{X} \hat{\boldsymbol{A}}_k \tag{A.7}$$

### A.1.2. Gaussian Process Regression

The output of the regressor $\boldsymbol{y}^*$ is to estimate a posterior distribution for any given new input data $\boldsymbol{x}^*$. The combined joint probability for the given data and new data can be given as:

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{y}^* \end{bmatrix} \sim \mathcal{N}\left( \begin{bmatrix} \boldsymbol{\mu}(\boldsymbol{x}) \\ \boldsymbol{\mu}(\boldsymbol{x}^*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K}(\boldsymbol{x},\boldsymbol{x}) & \boldsymbol{K}(\boldsymbol{x},\boldsymbol{x}^*) \\ \boldsymbol{K}(\boldsymbol{x}^*,\boldsymbol{x}) & \boldsymbol{K}(\boldsymbol{x}^*,\boldsymbol{x}^*) \end{bmatrix} \right) \tag{A.8}$$

The above joint probability can be used to obtain posterior distribution that results again a Gaussian distribution.

$$p(\boldsymbol{y}^* \mid \boldsymbol{y}) \sim \mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*) \tag{A.9}$$

Where,

$$\begin{aligned} \boldsymbol{\mu}^* &= \boldsymbol{\mu}(\boldsymbol{x}^*) + \boldsymbol{K}(\boldsymbol{x}^*,\boldsymbol{x})\,\boldsymbol{K}(\boldsymbol{x},\boldsymbol{x})^{-1}(\boldsymbol{y} - \boldsymbol{\mu}(\boldsymbol{x})) \\ \boldsymbol{\Sigma}^* &= \boldsymbol{K}(\boldsymbol{x}^*,\boldsymbol{x}^*) - \boldsymbol{K}(\boldsymbol{x}^*,\boldsymbol{x})\,\boldsymbol{K}(\boldsymbol{x},\boldsymbol{x})^{-1}\,\boldsymbol{K}(\boldsymbol{x},\boldsymbol{x}^*) \end{aligned} \tag{A.10}$$

The kernel function in the above equation is so chosen to give the similarity of the input points and such the posteriors are also strongly correlated to the inputs. RBF and periodic kernels are two examples of commonly used kernel functions to define the similarity or correlation between consecutive data inputs. Figure 2.5 illustrates the learned trajectory for a given input data(+). The solid line represents the mean of the gaussian and the gray area defines the variance in the posterior.

The computational complexity of GP is $\mathcal{O}(n^3)$ which is much higher than the LWR. The GP is further extended in [NP08] and [PMS13] to reduce the complexity.

### A.1.3. Gaussian Mixture Regression

#### A.1.3.1. Gaussian Mixture Model

The data set can be modelled into GMM of dimensionality D and multiple Gaussians K [CGB07]. Each Gaussian out of K can be represented by a Gaussian function with mean

$\mu_k$ and covariance matrix $\sigma_k$. And each Gaussian can possess a prior, which represents an initial likelihood of the trajectory, which is defined without considering any input data. Prior of Gaussian also characterizes the trajectory generation. Consider a data sample $\boldsymbol{x}_t$ at time step $t$ with mean $\mu_k$ and covariance $\sigma_k$ of the $k^{th}$ Gaussian in the GMM.

$$\boldsymbol{X}_t = \begin{bmatrix} \boldsymbol{x}_t \\ \boldsymbol{y}_t \end{bmatrix}, \quad \boldsymbol{\mu}_i = \begin{bmatrix} \boldsymbol{\mu}_i^x \\ \boldsymbol{\mu}_i^y \end{bmatrix}, \quad \boldsymbol{\Sigma}_i = \begin{bmatrix} \boldsymbol{\Sigma}_i^x \boldsymbol{\Sigma}_i^{xy} \\ \boldsymbol{\Sigma}_i^{xy} \boldsymbol{\Sigma}_i^y \end{bmatrix} \tag{A.11}$$

In order to extract trajectory during reproduction, We estimate a multimodal distribution $\mathcal{P}(\boldsymbol{y_t}|\boldsymbol{x_t})$ at every time step.

$$\mathcal{P}\left(\boldsymbol{y}_t \mid \boldsymbol{x}_t\right) = \sum_{i=1}^{K} h_i\left(\boldsymbol{x}_t\right) \mathcal{N}\left(\hat{\boldsymbol{\mu}}_i^y\left(\boldsymbol{x}_t\right), \hat{\boldsymbol{\Sigma}}_i^y\right) \tag{A.12}$$

Where,

$$\hat{\boldsymbol{\mu}}_i^y\left(\boldsymbol{x}_t\right) = \boldsymbol{\mu}_i^y + \boldsymbol{\Sigma}_i^y \boldsymbol{\Sigma}_i^{x-1}\left(\boldsymbol{x}_t - \boldsymbol{\mu}_i^x\right) \tag{A.13}$$

$$\hat{\boldsymbol{\Sigma}}_i^y = \boldsymbol{\Sigma}_i^y - \boldsymbol{\Sigma}_i^y \boldsymbol{\Sigma}_i^{x-1} \boldsymbol{\Sigma}_i^{xy} \tag{A.14}$$

$$h_i\left(\boldsymbol{x}_t\right) = \frac{\pi_i \mathcal{N}\left(\boldsymbol{x}_t \mid \boldsymbol{\mu}_i^x, \boldsymbol{\Sigma}_i^x\right)}{\sum_k^K \pi_k \mathcal{N}\left(\boldsymbol{x}_t \mid \boldsymbol{\mu}_k^x, \boldsymbol{\Sigma}_k^x\right)} \tag{A.15}$$

#### A.1.3.2. Hidden Markov Model

Consider a simple first order Markov model [CL18]:

$$\mathcal{P}\left(s_1, s_2, \ldots, s_T\right) = \mathcal{P}\left(s_1\right) \prod_{t=2}^{T} \mathcal{P}\left(s_t \mid s_{t-1}\right) \tag{A.16}$$

Which can be represented as:

$$\mathcal{P}\left(s_t \mid s_1, s_2, \ldots, s_{t-1}\right) = \mathcal{P}\left(s_t \mid s_{t-1}\right) \tag{A.17}$$

initial state and transition matrix A can be defined by

$$\Pi_i = \mathcal{P}(s_1 = i) \; with \; \sum_{i=1}^{K} \Pi_i = 1 \tag{A.18}$$

$$a_{i,j} = \mathcal{P}(s_{t+1} = j | s_t = i) \tag{A.19}$$

Thus the Markov model can be represented as

$$\Theta^{\mathrm{MM}} = \left\{ \{a_{i,j}\}_{j=1}^{K}, \Pi_i \right\}_{i=1}^{K} \tag{A.20}$$

HMM differ from the above simple first-order Markov Model, where model comprises hidden states. HMMs can be considered as a Markov chain with hidden states or a GMM with latent variable. Similar to Markov model representation in the equation A.1.3.2 as follows:

$$\Theta^{\mathrm{HMM}} = \left\{ \{a_{i,j}\}_{j=1}^{K}, \Pi_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i \right\}_{i=1}^{K} \tag{A.21}$$

## A.2. Segmentation Algorithms

### A.2.1. Unsupervised Clustering

**Algorithm 1** Agglomerative Hierarchical Clustering [JWH13]

1: Begin with $n$ observations and measure (such as Euclidean distance) of all the $({}^{n}C_2) = n(n-1)/2$ pairwise dissimilarities.Treat each observation as its own cluster.
2: **for** $i = n, n-1, ...2$ : **do**
3: (a) Examine all pairwise inter-cluster dissimilarities among the i clusters and identify the pairs of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendogram at which the fusion should be placed.
4: (b) Compute the new pairwise inter-cluster dissimilarities among the i-1 remaining clusters.

### A.2.2. Change Point Based Segmentation

**Algorithm 2** CHAMP (Change Point Approximate Model Parameters) [NOA15]

**Input:** Observations $y1:n$, candidate models $q1, ..., qr$, prior distribution $\phi(q)$, min segment length $\alpha$, and max number of particles M

**Output:** Viterbi path of changepoint times and models

**procedure** ChangePointDetection($a, b$)
  $maxpath, prevqueue, particles = \{\}$ ▷ Initialize data structures
  $prev\_queue.push(1/r)$
  **for** $i = 1 : r$ **do**
    $new\_p = newParticle(pos = 0, model = q\_i, prev\_MAP = 1/r)$
    $particles.add(new\_p)$
  **for** $t = \alpha : n$ **do** ▷ Do for all incoming data, starting at time $\alpha$
    **if** $t >= 2\alpha$ **then** ▷ Add new particles
      $pref = prev\_queue.pop()$
      **for** $i = 1 : r$ **do**
        $new\_p = newParticle(pos = t - \alpha, model = q_i, prev\_MAP = rev)$
        $particles.add(new\_p)$
    **for** $p \in particles$ **do** ▷ Compute fit probabilities for all particles
      $p\_tjq = L(p.pos, t, q) \cdot p(q) \cdot p.prev\_MAP$
      $p.MAP = g(t - p.pos) \cdot p\_tjq$
    $max\_p = max_p p.MAP$ ▷ Find max particle and update Viterbi path
    $prev\_queue.push(max\_p.MAP)$
    $max - path.add(j = max\_p.pos, q = max\_p.model)$
    **if** $fparticles.length > M$ **then** ▷ Resample if too many particles
      $particles = stratOptResample(particles, M)$
  $v\_path = \{\}$ ▷ Recover the Viterbi path
  $curr\_cp = n$
  **while** $curr\_cp > 0$ **do**
    $j, q = max_p ath[curr_c p - a]$
    $v\_path.add(start = j, end = curr\_cp, model = q$
    $curr\_cp = j$
  **return** $v\_path$