
Robot Skill Representation, Learning and Control with Probabilistic Movement Primitives

**Fertigkeitsrepräsentation, Erlernen und Ausführung in der Robotik mittels
probabilistischen Bewegungsprimitiven**

Zur Erlangung des akademischen Grades Doktor-Ingenieur (Dr.-Ing.)
genehmigte Dissertation von Dipl.-Ing. Alexandros Paraschos aus Cholargos Attikis,
Griechenland

Tag der Einreichung: 6 Februar 2017, Tag der Prüfung: 26 Juni 2017
Darmstadt, 2017 – D 17

1. Gutachten: Prof. Dr. Jan Peters
2. Gutachten: Prof. Dr. Gerhard Neumann
3. Gutachten: Dr. Sylvain Calinon



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Fachbereich Informatik
Intelligent Autonomous Systems (IAS)
Computational Learning for
Autonomous Systems (CLAS)

Robot Skill Representation, Learning and Control with Probabilistic Movement Primitives
Fertigkeitsrepräsentation, Erlernen und Ausführung in der Robotik mittels probabilistischen Bewegungsprimitiven

Genehmigte Dissertation von Dipl.-Ing. Alexandros Paraschos aus Cholargos Attikis, Griechenland

1. Gutachten: Prof. Dr. Jan Peters
2. Gutachten: Prof. Dr. Gerhard Neumann
3. Gutachten: Dr. Sylvain Calinon

Tag der Einreichung: 6 Februar 2017

Tag der Prüfung: 26 Juni 2017

Darmstadt, 2017 – D 17

Bitte zitieren Sie dieses Dokument als:

URN: urn:nbn:de:tuda-tuprints-69474

URL: <http://tuprints.ulb.tu-darmstadt.de/id/eprint/6947>

Dieses Dokument wird bereitgestellt von tuprints,

E-Publishing-Service der TU Darmstadt

<http://tuprints.ulb.tu-darmstadt.de>

tuprints@ulb.tu-darmstadt.de



Die Veröffentlichung steht unter folgender Creative Commons Lizenz:

Namensnennung – Keine kommerzielle Nutzung – Keine Bearbeitung 4.0 International

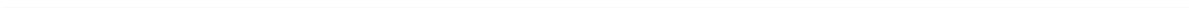
<http://creativecommons.org/licenses/by-nc-nd/4.0/>

Erklärung zur Dissertation

Hiermit versichere ich, die vorliegende Dissertation ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 8. November 2017

(Alexandros Paraschos)



Abstract

Robotic technology has made significant advances in the recent years, yet robots have not been fully incorporated in our every day lives. Current robots are executing a set of pre-programmed skills, that can not adapt to environmental changes, and acquiring new skills is difficult and time consuming. Additionally, current approaches for robot control focus on accurately reproducing a task, but rarely consider safety aspects that could enable the robots to share the same environment with humans. In this thesis, we develop a framework that allows robots to rapidly acquire new skills, to adapt skills to environmental changes, and to be controlled accurately and in a safe manner.

Our framework is based on movement primitives, a well-established concept for representing modular and reusable robot skills. In this thesis, we introduce a novel movement primitive representation that not only models the shape of the movement but also its uncertainty in time. We choose to rely on probability theory, creating a mathematically sound framework that is capable of adapting skills to environmental changes as well as adapting the execution speed online. Our probabilistic framework allows training the robot with imitation learning, speeding up significantly the process of novel skill acquisition. Hence, our approach unifies all the significant properties of existing movement primitive representations and, additionally, provides new properties, such as conditioning and combination of primitives.

By modeling the variance of the trajectories, our framework enables standard probabilistic operations to be applied on movement primitives. First, we present a generalization operator that can modify a given trajectory distribution to new situations and has improved performance over the current approaches. Secondly, we define a novel combination operator for the co-activating of multiple primitives, enabling the resulting primitive to concurrently solve multiple tasks. Finally, we demonstrate that smoothly sequencing primitives is simply a special case of movement combination. All aforementioned operators for our model were derived analytically.

In noisy environments, coordinated movements have better recovery from perturbations when compared to controlling each degree of freedom independently. While many movement primitive representations use time as a reference signal for synchronization, our approach, in addition, synchronizes complete movement sequences in the full state of the robot. The skill's correlations are encoded in the covariance matrix of our probabilistic model that we estimate from demonstrations. Furthermore, by encoding the correlations between the state of the robot and force/torque sensors, we demonstrate that our approach has improved performance during physical interaction tasks.

A movement generation framework would have limited application without a control approach that can reproduce the learned primitives in a physical system. Therefore, we derive two control approaches that are capable of reproducing exactly the encoded trajectory distribution. When the dynamics of the system are known, we

derive a model-based stochastic feedback controller. The controller has time-varying feedback gains that depend on the variance of the trajectory distribution. We compute the gains in closed form. When the dynamics of the system are unknown or are difficult to obtain, e.g., during physical interaction scenarios, we propose a model-free controller. This model-free controller has the same structure as the model-based controller, i.e. a stochastic feedback controller, with time-varying gains, where the gains can also be computed in closed form.

Complex robots with redundant degrees of freedom can in principle perform multiple tasks at the same time, for example, reaching for an object with a robotic arm while avoiding an obstacle. However, simultaneously performing multiple tasks using the same degrees of freedom, requires combining control signals from all the tasks. We developed a novel prioritization approach where we utilize the variance of the movement as a priority measure. We demonstrate how the task priorities can be obtained from imitation learning and how different primitives can be combined to solve unseen previously unobserved task-combinations. Due to the prioritization, we can efficiently learn a combination of tasks without requiring individual models per task combination. Additionally, existing primitive libraries can be adapted to environmental changes by means of a single primitive, prioritized to compensate for the change. Therefore, we avoid retraining the entire primitive library. The prioritization controller can still be computed in closed form.

Zusammenfassung

Obwohl die Robotik in den letzten Jahren erhebliche Fortschritte gemacht hat, sind Roboter noch nicht vollständig Teil unseres täglichen Lebens. Heutige Roboter führen eine Reihe vorprogrammierter Fähigkeiten aus, welche nicht an Änderungen in der Umgebung angepasst werden können, und das Erlernen neuer Fähigkeiten ist schwierig und zeitaufwendig. Zudem fokussieren sich aktuelle Ansätze der Robotersteuerung darauf, eine genaue Wiedergabe der Aufgabe zu erreichen, aber betrachten selten Sicherheitsaspekte, welche es Robotern ermöglichen würden, die gleiche Umgebung mit Menschen zu teilen. In dieser Arbeit entwickeln wir ein Framework, das Robotern erlaubt neue Fähigkeiten schnell zu erwerben, diese Fähigkeiten an Änderungen in der Umgebung anzupassen und genau auf sichere Art und Weise kontrolliert zu werden.

Unser Framework basiert auf Bewegungsprimitiven, ein etabliertes Konzept zur Repräsentation modularer und wiederverwendbarer Roboterfähigkeiten. In dieser Arbeit präsentieren wir eine neuartige Repräsentation von Bewegungsprimitiven, welche nicht nur die Form der Bewegung, sondern auch ihre Unsicherheit in der Zeit modelliert. Durch unsere Entscheidung, Wahrscheinlichkeitstheorie zu verwenden, auf haben wir ein mathematisch solides Framework kreiert welches imstande ist, Fähigkeiten an Änderungen in der Umgebung sowie die Ausführungsgeschwindigkeit online anzupassen. Unser probabilistischer Ansatz erlaubt das Trainieren des Roboters mittels Lernen durch Nachahmung, was das Erwerben einer neuen Fähigkeit dramatisch beschleunigt. Somit vereint unser Ansatz alle signifikanten Eigenschaften bereits existierender Repräsentationen von Bewegungsprimitiven und stellt zusätzlich weitere Eigenschaften zur Verfügung wie z. B. das Konditionieren und das Kombinieren von Primitiven.

Durch die Modellierung der Varianz der Trajektorien ermöglicht unser Framework die Anwendung von probabilistischen Operationen auf Bewegungsprimitive. Zunächst präsentieren wir einen Generalisierungsoperator der eine gegebene Trajektorienverteilung an eine neue Situation anpassen kann und verbesserte Leistungen auf bisherigen Ansätzen erzielt. Anschließend definieren wir einen neuartigen Kombinationsoperator zur Koaktivierung mehrerer Primitive, welcher dem resultierenden Primitiv ermöglicht gleichzeitig mehrere Aufgaben zu lösen. Zuletzt zeigen wir dass glattes Sequenzieren von Primitiven einfach ein Spezialfall des Bewegungskombinieren ist. Wir leiten alle zuvor genannten Operatoren für unser Modell analytisch her.

In verrauschten Umgebungen erholen sich koordinierte Bewegungen besser von Störungen im Vergleich zum unabhängigen Steuern eines jeden Freiheitsgrades. Während viele Ansätze für Bewegungsprimitive die Zeit als Referenzsignal zur Synchronisation verwenden, synchronisiert unser Ansatz zusätzlich komplette Bewegungssequenzen im ganzen Zustand des Roboters. Die Korrelationen der Fähigkeiten sind in der Kovarianzmatrix unseres probabilistischen Modells codiert, welches

wir anhand von Demonstrationen schätzen. Außerdem zeigen wir, dass durch das Codieren der Korrelationen zwischen dem Zustand des Roboters und Kraft-/Drehmomentsensoren unser Ansatz verbesserte Performanz in physischen Interaktionsaufgaben hat.

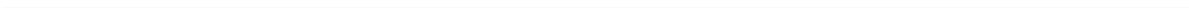
Ein Bewegungsgenerierungsframework hätte nur begrenzte Anwendungsmöglichkeiten ohne einen Steuerungsansatz um die gelernten Primitive auf einem physikalischen System zu reproduzieren. Deshalb leiten wir zwei Regelungsstrategien her, welche fähig sind, die codierte Trajektorienverteilung exakt zu reproduzieren. Für den Fall, dass die Dynamik des Systems bekannt ist leiten wir einen modellbasierten stochastischen Regler her. Dieser Regler hat zeitveränderliche Verstärkungsfaktoren, welche von der Varianz der Trajektorienverteilung abhängen. Wir berechnen die Verstärkungsfaktoren in geschlossener Form. Für den Fall, dass die Dynamik des Systems allerdings unbekannt oder schwer zu berechnen ist, z. B. in physikalischen Interaktionsszenarien, schlagen wir modellfreien Regler vor. Dieser modellfreie Regler hat die gleiche Struktur wie der modellbasierte Regler, d. h. es handelt sich um einen stochastischen Regler mit zeitveränderlichen Verstärkungsfaktoren, dessen Verstärkungsfaktoren ebenfalls in geschlossener Form berechnet werden können.

Komplexe Roboter mit redundanten Freiheitsgraden können grundsätzlich mehrere Aufgaben zur gleichen Zeit durchführen, wie zum Beispiel nach einem Objekt zu reichen und währenddessen ein Hindernis zu vermeiden. Jedoch erfordert die gleichzeitige Durchführung mehrerer Aufgaben welche die gleichen Freiheitsgrade beanspruchen das Kombinieren der Steuersignale aller Aufgaben. In unserem Framework haben wir einen neuartigen Priorisierungsansatz entwickelt, bei dem wir die Varianz der Bewegung als Maß für ihre Priorität verwenden. Wir zeigen, wie die Aufgabenprioritäten durch Nachahmungslernen gewonnen werden können und wie unterschiedliche Primitive kombiniert werden können, um auch zuvor unbeobachtete Aufgabenkombinationen zu lösen. Dank der Priorisierung kann unser Ansatz effizient eine Kombination von Aufgaben lernen, ohne dass einzelne Modelle für jede Aufgabenkombination erforderlich sind. Zusätzlich können bereits bestehende Bibliotheken von Primitiven an Änderung in der Umgebung angepasst werden, indem ein einziges Primitiv priorisiert wird, um die Änderung zu kompensieren. Dadurch vermeiden wir das erneute Trainieren der gesamten Bibliothek von Primitiven. Die Priorisierungssteuerung kann in geschlossener Form berechnet werden.

Acknowledgments

During the research and writing of this thesis I received an incredible amount of support and, therefore, I would like to thank:

- first and foremost, my thesis advisers Gerhard Neumann and Jan Peters for their support and constructive feedback throughout the developing of this thesis. Without your support, this thesis would not have been completed.
- Gerhard Neumann, to whom I am especially grateful, for introducing me to research, his advice, patience, and always being there to help me tackle any problem that arose in the last years.
- all my co-authors, Christian, Elmar, Marc, Rudolf, Simone, Hany, Peter, Daniel, Andras, Marco, Serena, and Oriane who through their feedback and discussions provided valuable insights for performing the research of this thesis.
- Sylvain Calinon, for the very interesting exchanges we had, for agreeing to be in my committee, and for his valuable and constructive input.
- my committee members, Professors von Stryk, Hollick, and Koepl, without whose time investment the defense would not have been possible.
- my colleagues for providing a pleasant working environment and for the many interesting and stimulating discussions. Especially, I would like to thank Elmar, Herke, Hany, and Oliver for their assistance and insights, and Filipe for thoroughly reviewing this thesis.
- Elmar, Filipe, Rudolf, and Simone, for their support, *listening* to my large or little problems and all the non-work related activities that provided the necessary breaks from research.
- my mother and Panagiota for providing me with unfailing support and encouragement throughout the years.
- Dimitris and Andreas for always being there for me.

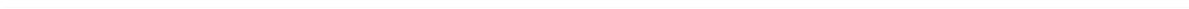


List of Symbols

The following table denotes the conventions and notation that we used throughout the thesis. Where possible, notation is kept consistent with prior work in the area.

Notation	Description
x	scalar
\mathbf{x}	vector
\mathbf{X}	matrix
\mathbf{X}^T	transpose of a matrix
\dot{x}	time derivative
$p(x)$	probability density

Symbols	Description
\mathbf{y}_t	state at time t
$\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}$	joint positions, velocities, and accelerations
$\mathbf{x}, \dot{\mathbf{x}}, \ddot{\mathbf{x}}$	Cartesian positions, velocities, and accelerations
\mathbf{u}_t	action at time t
Ψ_t	design matrix for a single DoF
$\Phi_t, \dot{\Phi}_t$	design matrix for all DoF and its time derivative
$\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t$	mean and covariance of the state at time t
$\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w$	mean and covariance of the weight distribution
$\mathbf{A}_t, \mathbf{B}_t, \mathbf{c}_t$	system matrix, control matrix, and drift vector
$\mathbf{K}_t, \mathbf{k}_t$	feedback gains matrix and feedforward vector
$\mathbf{J}, \dot{\mathbf{J}}$	Jacobian matrix and its time derivative tensor
\mathbf{J}^\dagger	pseudo-inverse of the Jacobian matrix
$\mathbf{M}, \mathbf{C}, \mathbf{G}$	mass, Coriolis, and gravity matrices



Contents

1	Introduction	1
1.1	Contributions	2
1.2	Thesis Outline	5
2	Probabilistic Movement Primitives	7
2.1	Introduction	7
2.2	Properties of Movement Primitive Frameworks	8
2.3	Related Work	10
2.4	Probabilistic Movement Primitives (ProMPs)	12
2.4.1	Probabilistic Trajectory Representation	13
2.4.2	Learning from Demonstrations	20
2.4.3	New Probabilistic Operators for Movement Primitives	22
2.4.4	Using Trajectory Distributions for Robot Control	27
2.5	Experiments	32
2.5.1	7-link Reaching Task	32
2.5.2	Double Pendulum	34
2.5.3	Playing Astrojax	36
2.5.4	Robot Maracas	36
2.5.5	Robot Hockey	38
2.5.6	Simulated Table Tennis	39
2.6	Epilogue	42
3	Model-Free Probabilistic Movement Primitives	43
3.1	Introduction	43
3.2	Related Work	44
3.3	Model-Free Probabilistic Movement Primitives	45
3.3.1	Encoding the Time-Varying State-Action Distribution of the Movement	46
3.3.2	Imitation Learning for Model-Free ProMPs	48
3.3.3	Integration of Proprioceptive Feedback	48
3.3.4	Generalization with Conditioning	49
3.3.5	Robot Control with Model-Free ProMPs	49
3.3.6	Correction Terms for Non-Linear Systems	50
3.3.7	Time adaptation and mixture of primitives	51
3.4	Experimental Evaluation	53
3.4.1	Reproduction and Generalization of the Trajectory Distribution	53
3.4.2	Generalization to different initial positions using a mixture of primitives	54
3.4.3	Adaptation of the interaction forces	55

3.4.4	Non-Linear Quad-Link Pendulum	55
3.4.5	Adaptation to External Forces on the iCub	57
3.4.6	Repositioning a chemistry flask	57
3.5	Epilogue	59
4	Prioritization of Movement Primitives	63
4.1	Introduction	63
4.2	Related work	64
4.3	Probabilistic Prioritization of Movement Primitives	65
4.3.1	Encoding Task Accuracy from Demonstrations	66
4.3.2	Probabilistic Combination of Tasks	67
4.3.3	Extension to Multiple Tasks	68
4.3.4	Robust Trajectory Distribution Tracking	69
4.4	Relation to Optimization-Based Prioritization Schemes	70
4.4.1	Comparison to Strict Prioritization Approaches.	70
4.5	Experimental Evaluation	71
4.5.1	Data-Driven Task-Space Adaptation	72
4.5.2	Incorporation of External Control Laws	73
4.5.3	Increased data efficiency	74
4.5.4	Initiating Contacts during Reaching	74
4.5.5	Adaptation of a Movement Primitive Library	76
4.6	Epilogue	77
4.A	Including the Dynamics of the System	78
5	Conclusion and Future Work	81
5.1	Summary of Contributions	81
5.2	Discussion and Future Work	82
5.3	Outlook	85
	Bibliography	89
	Curriculum Vitæ	97

1 Introduction

Many novel complex robotic platforms have been proposed in the recent years, yet robots have not been fully incorporated in our every-day life. Most of these platforms operate in well-structured environments, that do not drastically change, and are designed to be safe for both robots and humans. Having robots that share the same workplaces with humans, i.e., in unstructured environments, is one of the major goals of current research. However, it is a challenging task. The few robots that are capable of operating in such environments can only perform primitive tasks and are far from being general purposed. Creating general purposed robots that operate in human workplaces, poses several challenges that expand over different fields, including artificial intelligence, motor skills and vision. We focus on motor skill learning, particularly, the movement acquisition and generation problems.

Modeling, learning, executing, and adapting movements are required properties for robots to operate in unstructured environments. Flexible models that can represent highly complex skills are needed, enabling the robot to successfully perform in such environments. Facing a great variety of complex tasks, the robot should also be able to rapidly acquire new skills. In order to build flexible models rapidly, human demonstrations can be used as source of expert knowledge to bootstrap skill acquisition, (Ijspeert, 2008; Calinon et al., 2010a; Calinon, 2016).

Adapting the movements to new situations enable the robot to compensate for deviations in the environment, or to fulfill user preferences, (Ijspeert et al., 2013; Calinon, 2016). For example, reaching for an object that is not always placed in the exact same location, or, in a house-assisting scenario, turning on the lights while adapting the skill to match the specific switch. Adaptation is critical, but to adapt the skill to a new situation, the amount of change with respect to the original, non-adapted skill, should be kept to a minimum. By not completely changing the original skill, we assure that we maintain any implicit knowledge encoded. For example avoiding obstacles in the path of the robot's end-effector, could be implicit encoded in the shape of the movement. We demonstrate that in order to adapt in unstructured environments, it is necessary to use the skills uncertainty. Therefore, the uncertainty needs to be incorporated into the model during learning.

Robots with a flexible representation of skills, but lacking a control approach to reproduce them, have limited application. The robot should be able to successfully reproduce the task at hand, and simultaneously be compliant to avoid damaging the environment or itself. For this purpose, the robot could utilize a model of both its and the environment's dynamics to ensure that the task is reproduced successfully, while maintaining a degree of compliance. In this case, we can once more rely on the uncertainty of the skill to modulate the compliance of the robot during reproduction, (Calinon et al., 2010b).

Complex skills often involve interacting with objects of unknown dynamics. For example, interactions with a novel objects, or objects with dynamics that are difficult

to model. A glass of water, where the water level can not be estimated accurately, or a box that slides on a surface with unknown friction coefficients, are examples of such interactions. To overcome the lack of accurate models, we explore model-free approaches that associate the current state of the robot to the desirable action. Therefore, we avoid learning explicitly the dynamics model of the system. To learn such models, the robot actions must be observed during demonstrations.

Performing complex dexterous movements requires the learned skills to be smoothly sequenced, interrupted, or activated simultaneously. For example, a humanoid robot should not stop between walking and running. Instead, the two skills should be blend together, creating a smooth transition. Generally, during the execution of any skill another skill might become more appropriate. Additionally, complex robots with redundant degrees of freedom can in principle perform multiple tasks at the same time. For example, reaching for an object with a robotic arm while avoiding an obstacle. Simultaneously performing multiple tasks using the same degrees of freedom, requires combining the control signals from all tasks. Current approaches (Peters et al., 2007; Khatib, 1987; Khatib et al., 2004; Nakamura et al., 1987; Sentis and Khatib, 2005; Park et al., 2002; Bruyninckx and Khatib, 2000; Luh et al., 1980; Baerlocher and Boulic, 1998) resolve the movement combination problem by assigning task priorities. In contrast to current approaches, where task priorities are manually set, a principled way of learning the priorities from demonstrations is preferred, in order to minimize expert intervention.

Summing up, in this thesis our aim is to produce a novel movement primitive framework capable of modelling complex skills while representing the skills uncertainty. Skills should be learned from human demonstration and adapted to novel situations. We wish to derive control approaches with time-varying gains that can accurately reproduce the modeled skills on physical systems. Our framework should be able to handle physical interaction tasks, where system dynamics are unknown. Finally, the framework should promote solving complex tasks by combining multiple skills.

1.1 Contributions

In this section, we briefly discuss the main contributions of the thesis and we compare them to the state-of-the-art methods for motor control. We present the probabilistic modelling of movement primitives, we continue with movement primitive suitable for physical interaction tasks, and we conclude by presenting a novel probabilistic scheme for combining multiple skills.

Probabilistic Modeling of Movement Primitives

Movement Primitives (MP) are a well-established approach for representing modular and re-usable robot movement generators that can be composed into complex movements (Schaal et al., 2003a; Neumann et al., 2009; Rückert et al., 2012; Rozo et al., 2013). They provide an easy-to-learn representation of the primitive and are the key of recent imitation and reinforcement learning successes (Kober et al., 2010; Kormushev et al., 2010; Daniel et al., 2012). Current MPs approaches offer viable properties,

Table 1.1: An overview of the contributions of the thesis and the corresponding chapters that the contributions are presented or used.

	Chapter 2	Chapter 3	Chapter 4
Probabilistic Modeling of MPs	✓	✓	✓
Adaptation	✓	✓	○
Combination	✓	○	✓
Model-Based Controller	✓	○	✓
Model-Free Controller	○	✓	○
Prioritization of Tasks	○	○	✓

such as concise representations of the inherently continuous and high dimensional space of robot movements, generalization capabilities to novel situations (Ijspeert et al., 2003; Calinon et al., 2014; Calinon, 2016), temporal modulation of the primitive, sequencing of primitives, coupling between the degrees of freedom of the robot, and controllers for real time execution (Calinon et al., 2010a; Calinon, 2016). However, no single MP framework exists that offers all these properties. We developed a new representation of MPs that enables the unification of all the properties in a single framework. Our representation models, instead of a single trajectory (Ijspeert et al., 2003), a distribution of typically stochastic movements and captures its variance. It enables the derivation of new operators on MPs, including the generalization to new situations, the combination of multiple primitives, and the derivation of stochastic feedback control laws, in contrast to other probabilistic approaches to MPs (Calinon and Billard, 2009; Khansari-Zadeh and Billard, 2011) which do not support all operations.

Combination and Sequencing of Primitives

We take advantage of the probabilistic representation of the movement primitives to introduce novel operators to smoothly change or activate multiple primitives. First, we develop a co-activation operator that can either enable multiple primitives at the same time or smoothly sequence primitives. We show that both operations are special cases of the same mathematical formulation, and we derive them analytically (Paraschos et al., 2013a). In contrast to other MP approaches (Muelling et al., 2010; Matsubara et al., 2011; Forte et al., 2012), the co-activation operation of the primitives can solve multiple tasks *concurrently*. A detailed presentation of the new operators for MPs can be found in Chapter 2.

Model-Based Reproduction of the Trajectory Distribution

To reproduce the encoded skills on the physical system, we derive a stochastic model-based linear feedback controller with time-varying gains capable of reproducing exactly the encoded variability of the movement. The controller uses the variable gains at every time-step during the execution of the movement to match the variance of the trajectory distribution. The controller is capable of reproducing accurately the coupling among the degrees of freedom of the robot. For non-linear systems, a linearization of the system's model is computed in real-time using first-order Taylor expansion. All derivations of the stochastic feedback controller are performed in closed-form. We evaluate the proposed controller in a variety of simulated and real-robot tasks.

Probabilistic Movement Primitives for Physical Interaction Tasks

Physical interaction requires robots to accurately follow kinematic trajectories while modulating the interaction forces and torques to accomplish the task at hand and to be safe to the environment. However, current approaches rely on accurate physical models or use iterative learning approaches to accomplish the task. We present a versatile approach for physical interaction tasks, that can learn physical interaction tasks solely by demonstrations, without explicitly modelling the robot or the environment. We present an extension to our framework that allows accurate reproduction of the skill without modelling the system dynamics and, further, we incorporate external sensors, as for example, force/torque sensors. We derive a variable-stiffness controller in closed form that not only reproduces the trajectory and interaction forces present in the demonstrations, or adapts them to user's input.

Prioritized Combination of Movement Primitives

Movement prioritization is a common approach to combine controllers of different tasks for redundant robots, where each task is assigned a priority. The priorities of the tasks are often hand-tuned or the result of an optimization, but seldomly learned from data. Our framework develops a probabilistic prioritization, where the skill's priority is related to its uncertainty. We use our approach to prioritize full motion sequences and solve complex tasks. We demonstrate how the task priorities can be obtained from imitation learning and how different primitives can be combined to solve even unseen task-combinations. Due to the prioritization, our approach can efficiently learn a combination of tasks without requiring individual models per task combination. Further, our approach can adapt an existing primitive library by prioritizing additional controllers, for example, for implementing obstacle avoidance. Hence, the need of retraining the whole library is avoided in many cases.

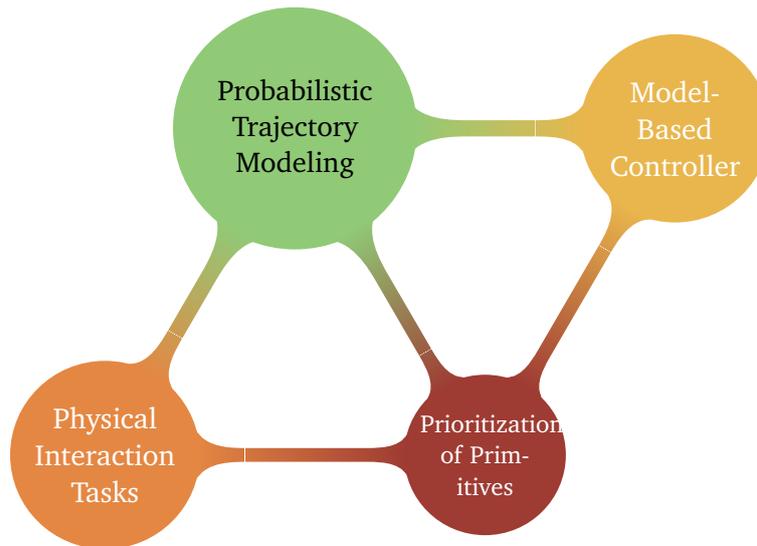


Figure 1.1: The core concepts presented in the thesis. We build around the Probabilistic Trajectory Modeling implementing a model-based controller that can reproduce accurately the learned trajectory distribution. We expand our framework with a model-free control approach, that is better suited for physical interaction. Finally, we introduce a probabilistic approach for task combination, where the activation of each task is learned from demonstrations.

1.2 Thesis Outline

The chapters of this thesis are mostly structured to be independent from each other, in order to ease reading. At the beginning of each chapter, we provide a brief recap of the necessary theory. We begin by introducing the novel movement primitive representation, the Probabilistic Movement Primitives in Chapter 2, the core approach on which this thesis is built. In Chapter 3 we focus on tasks that involve the robot physically interacting with the environment, or when the system dynamics are unknown. In Chapter 4 we combine primitives to solve simultaneously multiple tasks. In Chapter 5, we summarize the main contributions of this thesis and discuss the advantages and disadvantages compared to the state-of-the-art. We conclude with open challenges and potential improvements of probabilistic movement representations for skill learning and control.

Chapter 2 presents the Probabilistic Movement Primitives and their properties. It introduces the core modelling approach that we follow through this thesis. Additionally, in this chapter, we present how a Model-Based stochastic feedback controller can be derived analytically. The controller is capable to exactly reproduce the encoded trajectory distribution.

Chapter 3 presents a new control approach suitable for physical interaction, when the dynamics of the system are not known, or too complex to be modeled. In this approach we jointly learn the trajectory distribution and the appropriate control sig-

nals to reproduce it. We derive a Model-Free stochastic controller with time-varying gains that is capable of reproducing accurately the learned trajectory distribution. We evaluate our approach in linear and non-linear systems, that physically interact with the environment.

Chapter 4 introduces a new approach for combining movement primitives. The combination of primitives is data-driven, as the task priorities are obtained from imitation learning. We show that multiple primitives can be combined to solve even unseen task-combinations. Our approach can efficiently learn a combination of tasks. Further, by combining primitives, can adapt an existing primitive library to changes in the environment, e.g., when an obstacle is introduced, without retraining the library. The resulting hierarchical controller is computed in closed form.

Chapter 5 summarizes our approach and presents the main conclusions of this thesis. Further, we discuss open challenges and the potential extensions of our approach.

2 Probabilistic Movement Primitives

2.1 Introduction

Movement Primitives (MPs) are a well-established approach for representing movement policies in robotics. MPs have several beneficial properties; generalization to new situations, temporal modulation of the movement, co-activation of multiple primitives to concurrently solve multiple tasks, sequencing of primitives to generate longer and more complex movements, and they are easy to learn from demonstrations. Using such properties, MPs were successfully applied to reaching (dAvella and Bizzi, 2005), locomotion (Dominici et al., 2011; Moro et al., 2012) and are state of the art for robot movement representation and generation. However, many approaches for movement generation based on MPs (Ijspeert et al., 2003; Williams and Storkey, 2007; dAvella and Bizzi, 2005; Khansari-Zadeh and Billard, 2011; Rozo et al., 2013; Rückert et al., 2012; Righetti and Ijspeert, 2006) exhibit only a subset of these properties. Hence, a generalized framework that unifies all these properties in one principled framework is needed.

We formalize the concept of probabilistic movement primitives (ProMPs) as a general probabilistic framework for representing and learning MPs. A ProMP represents a distribution over trajectories. The trajectory distribution can be defined in either joint-space, task-space, or any other space that accommodates the experiment. In this chapter, we focus on joint-space trajectories. Working with distributions enables us to formulate the described properties using operations from probability theory. For example, modulation of a movement to a novel target can be realized by conditioning on the desired target’s positions or velocities. Similarly, consistent parallel activation of two elementary behaviors can be accomplished by a product of two independent trajectory distributions. A trajectory distribution can encode the variance of the movement, and, hence, a ProMP can directly encode optimal behavior in systems with linear dynamics, quadratic costs and Gaussian noise (Todorov and Jordan, 2002). In contrast, deterministic approaches, e.g., the DMP approach, can only represent the mean solution, which is known to be suboptimal. Even if assumption does not hold, we believe that it offers a good approximation of physical robotic systems. Finally, a probabilistic framework allows us to model the coupling between the degrees of freedom (DoFs) of the robot by estimating the covariance between different DoFs.

The benefits of using a probabilistic representation have so far not been extensively exploited for representing and learning MPs. The main reason for this limitation has been the difficulty of extracting a policy for controlling the robot from a trajectory distribution. We show how this step can be accomplished and derive a control policy that exactly reproduces a given trajectory distribution. While ProMP introduces many novel components, it also incorporates many of the advantages from well-known pre-

vious movement primitive representations (Schaal et al., 2003b; dAvella and Bizzi, 2005), such as temporal rescaling of movements and the ability to represent both rhythmic and stroke based movements.

Further, we introduce a new regularization technique for achieving smoother movements and present an expectation-maximization algorithm for learning rhythmic ProMPs in more detail. We extended the description of our controller derivation and show how it is used on physical tasks, e.g. controlling a 7-DoF arm for playing Maracas, robot-hockey, and ‘Astrojax’. Moreover, we show new comparisons to state of the art MP approaches in terms of optimality, generalizability, composition of primitives and robustness of the movement representations. We also evaluate our ProMP controller on non-linear systems and made the source code of all examples publicly available¹.

2.2 Properties of Movement Primitive Frameworks

We categorize MPs into state-based (Khansari-Zadeh and Billard, 2011; Calinon et al., 2010a) and trajectory-based representations (Schaal et al., 2003b; Neumann et al., 2009; Rückert et al., 2012; Rozo et al., 2013). Trajectory-based primitives typically use time as the driving force of the movement. They require simple, typically linear, controllers, and scale well to a large number of DoFs. In contrast, state-based primitives (Khansari-Zadeh and Billard, 2011; Calinon et al., 2010a) do not require the knowledge of a time step but often need to use more complex, non-linear policies. Such increased complexity has limited the application of state-based primitives to a rather small number of dimensions, such as the Cartesian coordinates of the task space of a robot. The main focus of this chapter is on trajectory-based representations. We begin with a discussion on the properties of MPs.

Concise representation. MPs offer a concise representation of the movement, with a few open parameters to set. The small number of parameters simplifies learning the movement from demonstrations and the use of reinforcement learning algorithms to adapt and refine the primitive through trial-and-error. MP frameworks can be trained from demonstrations using simple learning methods, e.g. linear regression, and have been successfully used in fairly complex scenarios, including “Ball-in-the-Cup” (Kober et al., 2010), Ball-Throwing (Ude et al., 2010; da Silva et al., 2012), Pancake-Flipping (Kormushev et al., 2010), Tetherball (Daniel et al., 2012), and bipedal locomotion (Nakanishi et al., 2004).

Adaptation and time modulation. Many MPs offer an intrinsic adaptation mechanism to match a new situation or an altered task, e.g., hitting a different incoming balls when playing table tennis. The adaptation commonly comes in a form of modification of the desired target position and velocity at the end of the primitive or as a modulation of the amplitude of the primitive (Ijspeert et al., 2003). Our approach (Paraschos et al., 2013a,b) can be used to adapt the movement at any time point during the trajectory’s execution.

¹ http://www.ausy.tu-darmstadt.de/uploads/Team/AlexandrosParaschos/ProMP_toolbox.zip

Furthermore, adaptation of MPs include temporal modulation. Temporal modulation is a valuable property as it enables MPs to be applied in scenarios where correct timing is critical for the success of the task, e.g., in hitting, batting, or in locomotion to adjust the walking speed of the robot (Righetti and Ijspeert, 2006).

Combination and sequencing. The expressiveness of an MP approach can be significantly improved if multiple primitives can be simultaneously co-activated to compose more complex movements. However, most MP approaches do not support co-activation of primitives in a principled way. Instead, the concurrent activation requires a prioritization scheme (Mülling et al., 2013; Pastor et al., 2011) in order not to disrupt the motion. In our approach (Paraschos et al., 2013a), we co-activate primitives to solve multiple tasks at the same time, without the need of such a scheme. Besides simultaneous activation, MP architectures aim to support sequencing MPs (Konidaris et al., 2012) to acquire a smooth transition from one primitive to another. Such sequencing is needed to dynamically concatenate primitives in order to acquire longer, more complex movements. We show that in our framework a smooth transition can be achieved in a principled way similar to the combination of primitives.

Coupling the DoFs. Movement primitives approaches are typically applied to robots with multiple Degrees of Freedom (DoF). In order to reproduce coordinated movements, MPs need a synchronization mechanism among the different DoF. Using time, or a function of time, as a reference signal (Schaal et al., 2007; Ijspeert, 2008), one can implement simple time alignment mechanisms. However, when experiencing deviations from the desired trajectory due to noise or unmodeled effects, coordinated recovering from perturbations is advantageous. ProMPs, additionally to time synchronization, estimate such correlations directly from demonstrations and use them to synchronize the DoFs of the system.

Optimal behavior. Many trajectory-based representations use a single desired trajectory that is followed by a feedback controller with constant gains. However, following such a single trajectory has been proven to be suboptimal for many tasks if the system’s dynamics are stochastic (Todorov and Jordan, 2002). In this chapter, we focus on control affine systems with Gaussian control noise, which is a standard assumption for physical systems. In this case, a distribution over trajectories is a good representation of the optimal behavior. Such distribution can be achieved by using time-varying feedback gains, which are often used as approximation for optimal behavior (Li and Todorov, 2004). Feedback controllers with time varying gains modulate the stiffness of the system to provide high precision at the ‘important’ time points of a task while the system is less controlled for time points where accurate control is not so critical. The time varying gains of the controller can be approximated (Calinon et al., 2010b), computed using a LQR by specifying a cost function (Calinon, 2016; Bruno et al., 2015), improved with reinforcement learning (Buchli et al., 2011), or, as in our approach, computed in closed form (Paraschos et al., 2013a).

Stability. Generating stable behavior is an important aspect of MPs. However, stability guaranties often have limited use as they assume linearity in the dynamics. Yet, however simple, real-world, systems are non-linear, e.g., a pendulum, where the gravity alone introduces non-linearities in the dynamics. Discrete DMPs (Ijspeert et al., 2003) generate stable behavior by moving towards an attractor at the end of the movement, while periodic MPs (Ijspeert et al., 2003; Righetti and Ijspeert, 2006)

stabilise the movement on a unit circle. The probabilistic framework from (Calinon et al., 2010a) initially did not provide any stability guarantees, but it was still generating stable movements as long as the disturbances did not perturb the system “far” from the region where the demonstration occurred. With (Khansari-Zadeh and Billard, 2011) the authors alleviate the problem and learned asymptotically stable control laws. Recently, Calinon et al. (Calinon, 2016) proposed the use of a Linear Quadratic Regulator (LQR) for control, that is stable for closed-loop systems (Stengel, 2012). The ProMPs (Paraschos et al., 2013a) derive a controller that exactly reproduces the demonstrated trajectory distribution and, thus, provide stability guaranties as long as the demonstrated trajectory distribution was generated by a stable control law.

2.3 Related Work

A commonly used trajectory-based representation is the Dynamic Movement Primitive (DMP) approach, introduced in (Ijspeert et al., 2003) and (Ijspeert et al., 2013) for a recent review. They represent a linear attractor system which is modulated by a time-dependent forcing function. The DMP introduced the concept of a phase, defined as a monotonic function of time. By adjusting the phase derivatives, we can temporally scale the movement. The forcing function is represented by normalized Gaussian basis functions, multiplied with the phase signal. Since the phase decreases exponentially to zero, the forcing function will asymptotically vanish at the end of the movement. At that time, only the attractor dynamics stay active, which guarantees the stability of the linear system. When used in an imitation learning scenario, the weights of the basis functions can be fitted from a single demonstration using linear regression. Generalization to new, unseen, situations in DMPs is limited. The original formulation only allowed for changing the position at the end of the movement, which is implemented by modifying the position of the goal attractor or, for rhythmic DMPs, by adjusting the amplitude of the forcing function. Extensions exist that also allow setting a desired final velocity (Kober et al., 2010; Mülling et al., 2013; Pastor et al., 2009). Directly changing intermediate points in the trajectory is not possible. DMPs can be sequenced given proper initialization (Pastor et al., 2009), but only instant switching from one primitive to another is considered. (Kulvicius et al., 2012) extended DMPs to support sequencing of primitives and evaluated their approach on a handwriting dataset. (Gams et al., 2014) proposed the use of DMPs for tasks that include interactions with the environment.

Despite that DMPs introduce many beneficial properties, such as temporal scaling of the movement, learning from a single demonstration or generalizing to new final positions, further work is still needed for concurrently activating multiple primitives, generalizing to intermediate via-points, representing optimal behavior in stochastic systems, and capturing the correlation of the individual joints of the robot. Trajectories based on DMPs applied to multiple DoF systems are synchronized based only on the internal phase variable. Multiple DMPs for the same DoF cannot be activated simultaneously without further considerations on prioritized control and partial cancellation of the movement.

Probabilistic approaches use distributions to additionally encode the variability of the movement (Calinon et al., 2010a; Rozo et al., 2013; Kormushev et al., 2010; Calinon, 2016). The variability of the movement, or the variance in distribution terms, is crucial, as it reflects the importance of single time points for the movement execution and it is often a requirement for representing optimal behavior in stochastic systems (Todorov and Jordan, 2002). Moreover, capturing the variance of the movement leads to better generalization capabilities and to more natural movements. A probabilistic MP approach was proposed by (Calinon et al., 2010a), where a Gaussian Mixture Regression (GMR) model was used to represent the trajectory. Given a set of trajectories, the GMR was trained with an Expectation Maximization (EM) algorithm (Roza et al., 2013). A unifying formulation that extends the DMPs and uses them in a probabilistic framework is discussed in (Kormushev et al., 2010). Yet, it is unclear how a GMR model can be conditioned to reach different final or intermediate positions. An extension of the approach (Calinon, 2016) enabled generalization to different situations by recording the movement from different spaces and tracking the affine transformation to each space. While the approach is capable of generalizing, for example when an object changes its position, it can not modulate the encoded variance.

Besides representing the variance of the trajectory, we need a controller that reproduces the encoded distribution on a real system. A feedback controller where the gains are based on the inverse of the covariance of the current time-step was presented in (Calinon et al., 2010b). The control law is based on the intuition that the gains have to be lower when the variance of the trajectories is higher. A comparison to this control law is presented at the evaluation section of this chapter. As our experiments show, the resulting trajectory distribution from executing this controller does not match the desired one. In (Calinon, 2016; Bruno et al., 2015), the authors proposed the use of minimum intervention control to generate the gains of the feedback controller. In this approach, the authors use the inverse of the covariance at every time point as metric for the quadratic state costs. However, while intuition-wise weighting the state with the inverse of the covariance is appropriate, we will show in our comparison that this approach can not match the desired trajectory distribution. Additionally, the cost function proposed by the authors include a quadratic action penalty to limit the actions that is not learned by the demonstrations.

A different approach for computing a control law for a GMR model was proposed by Khansari-Zadeh et al. (Khansari-Zadeh and Billard, 2011). In this approach, the control gains are proven to be stable if the system is linear. The authors derive the stability constraints from the Lyapunov stability theory. In (Khansari-Zadeh et al., 2014), the authors extend their approach to generate stable controllers with state-dependent stiffness. The resulting controller share similarities with the ProMPs controller.

The approach by (Rückert et al., 2012) also offers a probabilistic interpretation of MPs by representing them with learned graphical models. A probabilistic planning algorithm is used to obtain a controller that optimizes the cost function represented by the graphical model. The resulting controller is also a linear feedback controller with time varying gains. However, this approach heavily depends on the quality of the used planner and imitation learning of such a representation is not straightforward.

The ability to combine multiple MPs into a single movement provides significantly better generalization capabilities, enables the use of MP libraries, and has recently attracted attention of the community. (Mülling et al., 2013) use a library for table tennis which is concurrently activating multiple DMPs to perform striking movements. Each primitive is activated with an activation provided by a trained gating network. The primitives are then combined on the acceleration level which is equivalent to a linear combination of primitives in parameter space. The primitives and the activation weights were refined with Reinforcement Learning (RL). A different approach was proposed by (Matsubara et al., 2011) using DMPs in combination of with a style parameter. The parameters of DMPs are linearly interpolated according to the given style parameter. (Forte et al., 2012) proposed a similar approach, where a library of DMPs learned from multiple demonstrations is used. Generalization is obtained from a Gaussian Process Regression (GPR) model which is capable of modeling non-linear transformations of the style variable. The major limitation of approaches based on deterministic representations, e.g., on DMPs, is the inability to concurrently solve a combination of tasks where we have one task per primitive. Since there is no notion of the importance of each time point in the trajectory the resulting combined primitive is just an interpolation of the participating primitives trajectories. In contrast, probabilistic representations (Khansari-Zadeh and Billard, 2011; Calinon et al., 2010a) leave unclear how primitives can be combined. In ProMPs, we propose a new combination operator based on a product of trajectory distributions. We show that by co-activating ProMPs, the resulting movement solves a combination of tasks that is given by a combination of different cost functions. We evaluate this property in two different scenarios in the experiments section.

Smoothly sequencing, also called blending, two movement primitives can be considered as a special case of a combination of MPs. Discrete DMPs can be trivially sequenced (Pastor et al., 2009), however the transition from one primitive to then next one is typically instantly, which can lead to a jump in the acceleration profiles. Special cases of discrete and periodic primitive blending, such as transient motions, have been considered in (Ernesti et al., 2012; Degallier et al., 2011). As opposed to the previous approaches, the ProMPs can cope with combination and blending of primitives independently of their periodicity.

In the next section, we will first introduce probabilistic movement primitives and show their advantageous properties. Next, will show how to compute a time-varying feedback controller that reproduces the given trajectory distribution. Subsequently, we will demonstrate the performance and advantageous properties of ProMPs in several experiments on simulated and real robot tasks.

2.4 Probabilistic Movement Primitives (ProMPs)

ProMPs provide a single principled framework for implementing the desirable properties of MPs, summarized in Table 2.1. We will first introduce the probabilistic model for representing the trajectory distribution, that is based on a basis function representation. Such representations significantly reduces the amount of model parameters and facilitates learning. We proceed by illustrating how our representation can be trained from imitation data for both stroke-based and periodic movements. Training

Table 2.1: Properties and their implementation in the ProMPs

Property	Implementation
Co-Activation	Product of $p_i(\boldsymbol{\tau})$
Modulation	Conditioning
→ final positions	✓
→ final velocities	✓
→ via-points	✓
Optimality	Encode variance
Coupling	Mean, Covariance
Learning	Max. Likelihood
Temporal Modulation	Modulate Phase
Rhythmic Movements	Periodic Basis

from imitation allows to rapidly reproduce tasks that are easy to demonstrate to the robot. Here, we describe a simple maximum likelihood training procedure that can be used for stroke-based movements and an expectation-maximization algorithm that can be used to train the primitive in case of missing data or also for rhythmic movements. We continue by discussing the implementation of the desirable properties, i.e. temporal modulation of the movement, encoding of the coupling between the joints that allows the generation of coordinated movements, conditioning to generalize a trained primitive to a novel situation, adaptation to task parameters to allow task-dependent variables to modify the primitive, and combination and blending of primitives to solve more complex tasks. Finally, in Sec. 2.4.4, we present the analytical derivation of a stochastic feedback controller that is capable of exactly reproducing the trajectory distribution. Such feedback controller is essential for using trajectory distributions for controlling a physical system.

2.4.1 Probabilistic Trajectory Representation

We begin our discussion with the simple case of a single degree of freedom, where the joint angle q is a scalar, and we subsequently extend it to the multiple DoF case, where the vector \mathbf{q} describes multiple joint angles. We model a single movement execution as a trajectory $\boldsymbol{\tau} = \{q_t\}_{t=0..T}$, defined by the joint angle q_t over time. In our framework, a MP describes multiple ways to execute a movement, which naturally leads to a probability distribution over trajectories. We encode our policy representation with a hierarchical Bayesian model, which is presented in Figure 2.1.

Concise Encoding of Trajectory Distributions

Our movement primitive representation models the time-varying variance of the trajectories. Representing the variance information is crucial as it reflects the importance of single time points for the movement execution. We use a basis-function representation as it reduces the amount of model parameters in comparison to a simple

distribution over the joint positions for each time step. This reduction in parameters can greatly facilitate learning. Additionally, it allows us to derive a continuous time approach and transfer data between systems, e.g., from a motion capture system to the robotic platform, directly without interpolating the data. When controlling the system, a continuous time approach allows for choosing the control frequency and is robust to jitter. Further, as we will discuss in Sec. 2.4.1, it enables the temporal modulation of the movement. Additionally, it allows us to generalize the primitive at any time-point, Sec. 2.4.3 and to derive our feedback controller in closed form, Sec. 2.4.4.

We use a weight vector \mathbf{w} to compactly represent a single trajectory. The probability of observing a trajectory $\boldsymbol{\tau}$ given the underlying weight vector \mathbf{w} is given as a linear basis function model

$$\mathbf{y}_t = \begin{bmatrix} q_t \\ \dot{q}_t \end{bmatrix} = \Phi_t \mathbf{w} + \epsilon_y, \quad (2.1)$$

$$p(\boldsymbol{\tau}|\mathbf{w}) = \prod_t \mathcal{N}(\mathbf{y}_t | \Phi_t \mathbf{w}, \Sigma_y), \quad (2.2)$$

where $\Phi_t = [\phi_t, \dot{\phi}_t]^T$ defines the $2 \times n$ dimensional time-dependent basis function matrix for the joint positions q_t and velocities \dot{q}_t . The basis functions for the velocities $\dot{\phi}_t$ are the time derivatives of ϕ_t . The variable n defines the number of basis functions and $\epsilon_y \sim \mathcal{N}(\mathbf{0}, \Sigma_y)$ represents zero-mean i.i.d. Gaussian noise.

In order to capture the variance of the trajectories, we introduce a distribution $p(\mathbf{w}; \boldsymbol{\theta})$ over the weight vector \mathbf{w} , with parameters $\boldsymbol{\theta}$. In most cases, the distribution $p(\mathbf{w}; \boldsymbol{\theta})$ will be Gaussian where the parameter vector $\boldsymbol{\theta} = \{\boldsymbol{\mu}_w, \Sigma_w\}$ specifies the mean and the variance of \mathbf{w} . However, also more complex distributions such as Gaussian mixture models can be used for this task (Rueckert et al., 2015a). The trajectory distribution $p(\boldsymbol{\tau}; \boldsymbol{\theta})$ can now be computed by marginalizing out the weight vector \mathbf{w} , i.e.

$$p(\boldsymbol{\tau}; \boldsymbol{\theta}) = \int p(\boldsymbol{\tau}|\mathbf{w})p(\mathbf{w}; \boldsymbol{\theta})d\mathbf{w}, \quad (2.3)$$

to obtain the probability distribution over the trajectories $\boldsymbol{\tau}$. The distribution $p(\boldsymbol{\tau}; \boldsymbol{\theta})$ defines the hierarchical Bayesian model that is illustrated at Figure 2.1. The model's parameters are given by the observation noise variance Σ_y and the parameters $\boldsymbol{\theta}$ of the weight distribution $p(\mathbf{w}; \boldsymbol{\theta})$.

Illustrative example. To illustrate the properties of our MP representation, we use a simple toy-task as a running example throughout this section where we also compare to other state-of-the-art MP approaches. In our toy-task, we use a trajectory distribution that passes through two via-points. The simulated system has linear dynamics and Gaussian i.i.d. noise on the actions. In this illustrative example, we

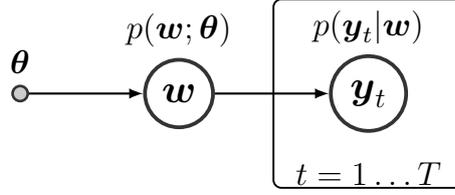


Figure 2.1: The Hierarchical Bayesian Model used in ProMPs. The probability distribution $p(\mathbf{y}_{1:T}|\mathbf{w})$ of the observed trajectories depends on the parameter vector \mathbf{w} . The distribution over the parameter vector \mathbf{w} is given by $p(\mathbf{w}|\boldsymbol{\theta})$. The parameter vector \mathbf{w} is integrated out in the ProMP formulation.

control the acceleration of the system. We generate demonstrations with an optimal control algorithm (Toussaint, 2009). The cost function is given as

$$C(\boldsymbol{\tau}, \mathbf{u}) = \sum_{i=\{t_{\text{via}}\}} (\mathbf{y}_i^d - \mathbf{y}_i)^T \mathbf{Q} (\mathbf{y}_i^d - \mathbf{y}_i) + \sum_{i=1}^T \mathbf{u}_i^T \mathbf{R} \mathbf{u}_i, \quad (2.4)$$

where $t_{\text{via}} = \{0.4s, 0.7s\}$ is a set of the time-points for the via-points and \mathbf{Q}, \mathbf{R} are the state and action cost matrices, respectively. We simulate trajectories with the resulting controller to obtain the demonstrations. The demonstrations exhibit variability due to the noise of the system. The optimal trajectory distribution is presented in Figure 2.2(a).

The use of a cost-function enables us to quantify the quality of the resulting MP policies. The ProMP policy is capable of reproducing exactly the variance of the movement, as shown in Figure 2.2(b). For the trajectory reproduction of ProMPs, we used the controller that we describe in Sec. 2.4.4. Additionally, we evaluate the heuristic controller presented in (Calinon et al., 2010b), which computes the feedback gains inverse proportionally to the variance of the trajectory. The trajectory distribution of the inverse covariance controller does not match the demonstrated distribution, see Figure 2.2(c). The DMP approach uses constant feedback gains to follow a single trajectory, and, hence, can not adapt the variance of the resulting trajectory distribution. In Figure 2.2(d), we generated trajectory distributions for two different settings of the feedback gains to illustrate the resulting variances. We empirically optimized the gains for the inverse covariance controller and the DMPs using search. The average costs generated by each control law are shown in the upper part of Table 2.2. The ProMP achieve a similar cost to the optimal controller while all other controllers can not reproduce the optimal behavior.

Further, we compare our approach to (Calinon, 2016), where we fit the proposed Gaussian Mixture Model (GMM) to the demonstrations and then use Gaussian Mixture Regression (GMR) to derive the desired trajectory distribution. We present the fitted regression model in Figure 2.3a (blue). We generated trajectories using Minimum Intervention Control (Calinon, 2016) and we present the results in Figure 2.3a (red) where we jointly optimized for the number of mixture components and the action penalty. We also used the optimal number of components, but the same action

penalty as in the cost function used to generate the demonstrations (green). The resulting controller can not reproduce the given distribution.

Moreover, we evaluated our approach using simple Gaussian distributions and optimal control. At every time-step, we fit a Gaussian distribution over the state and we use it to set a quadratic cost function. The cost function has the form of Equation (2.4) where \mathbf{y}_i^d is set to the mean and Q to the inverse of the covariance. We optimize for the action penalty \mathbf{R} such that the true cost function we used to generate the data is minimized. We present our results in Table 2.2. This approach uses the same approach for deriving the controller as in (Calinon, 2016), but uses a simple Gaussian distribution to model each time-step instead of the state-defined GMR. Compared to ProMPs, the performance on the true cost function is worse as can be seen in the table. This approach also does not provide any generalization or modulation mechanism.

As another baseline, we fit a Gaussian distribution at every time-step on the state- action space. At reproduction, we condition the distribution of that time-step on the current state to obtain the action, which results in a linear Gaussian action policy. As the demonstrations have been generated by a time-dependent linear controller, the performance of this approach is close to optimal and similar to the ProMP controller as shown in Table 2.2. However, fitting a Gaussian distribution over the state-action requires the actions to be known during the demonstrations and, which limits the applicability of the approach to tele-operation setups. Similar to the optimal control approach from the previous paragraph, this approach does not provide any generalization mechanism.

Table 2.2: Comparison of different control approaches on a hand-specified cost function. As baseline, we compare the approaches to an optimal controller that maximizes the cost. The ProMPs can produce trajectories with a similar cost. The newly presented regularization scheme for the weights (jerk penalty, Sec. 2.4.2) achieves a slightly lower costs due to the smoother torque profiles produced by this approach.

	Control Approach	Average Cost
Reproduction	Optimal Controller	$2.07 \cdot 10^4 \pm 2.58 \cdot 10^2$
	Model-Free Gaus. Ctl	$2.25 \cdot 10^4 \pm 3.21 \cdot 10^2$
	ProMP Jerk Penalty	$2.29 \cdot 10^4 \pm 3.35 \cdot 10^2$
	ProMP Weight Reg.	$2.35 \cdot 10^4 \pm 3.25 \cdot 10^2$
	Opt. Ctl. — Gaus. Dist.	$3.37 \cdot 10^4 \pm 4.41 \cdot 10^2$
	GMM/GMR - Min Int.	$4.47 \cdot 10^4 \pm 7.25 \cdot 10^2$
	DMP	$5.16 \cdot 10^4 \pm 13.2 \cdot 10^2$
	Inv. Cov. Controller	$7.36 \cdot 10^4 \pm 16.1 \cdot 10^2$
	DMP with Low Gains	$76.5 \cdot 10^4 \pm 392 \cdot 10^2$
Combin.	Optimal Controller	$3.36 \cdot 10^4 \pm 3.52 \cdot 10^2$
	ProMP	$5.46 \cdot 10^4 \pm 3.55 \cdot 10^2$
	Inv. Cov. Controller	$6.54 \cdot 10^4 \pm 7.30 \cdot 10^2$
	DMP	$208 \cdot 10^4 \pm 107 \cdot 10^2$

Temporal Modulation

With temporal modulation, we can adjust the execution speed of the movement. Similar to the DMP approach, we introduce a phase variable z to decouple the movement from the time signal. By modifying the rate of the phase variable, we can modulate

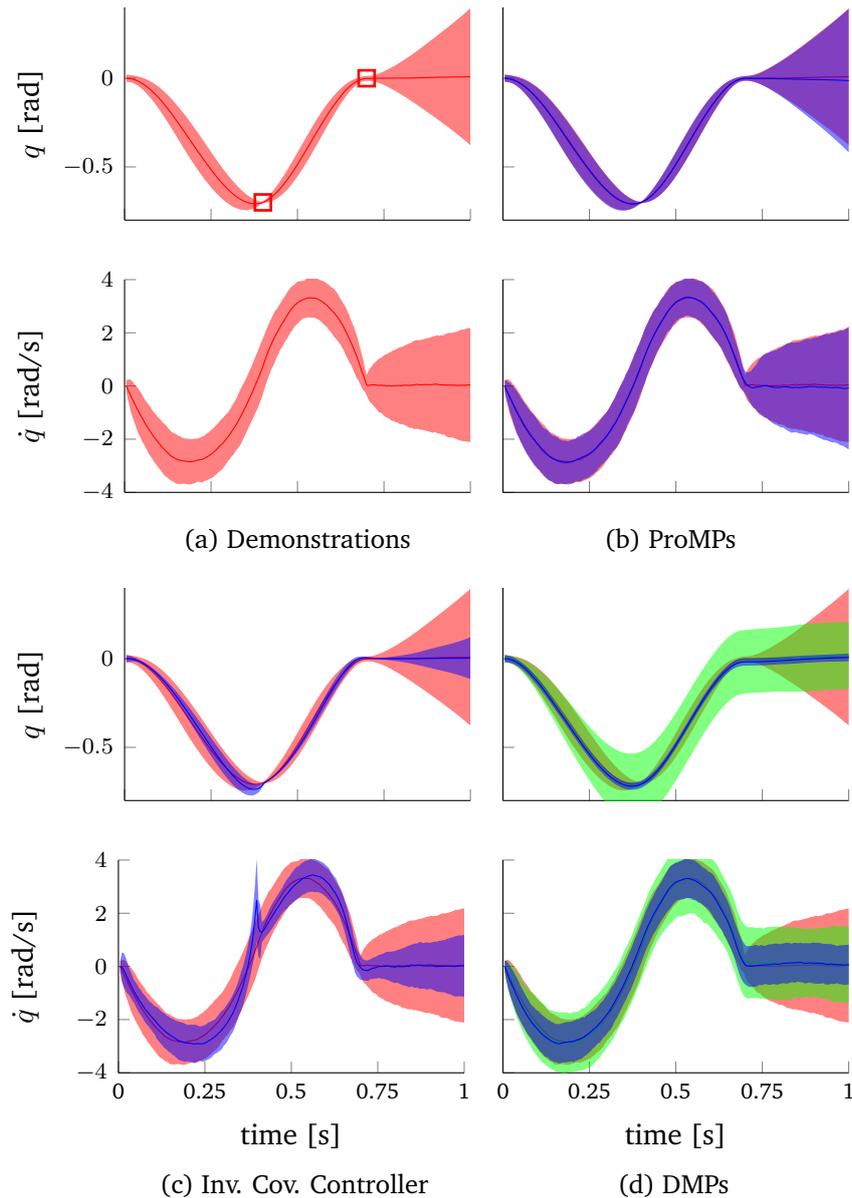


Figure 2.2: Trajectory distribution showing the joint positions (first row) and velocities (second row). The shaded area denotes two times the standard deviation. (a) The demonstrated trajectory distribution that was generated by a stochastic optimal control algorithm for a via-point task. The resulting trajectories show variability due to the noise in the system. (b) The trajectory distribution generated using ProMPs (blue). ProMPs can exactly reproduce the demonstrated trajectory distribution (shown in red below the blue shaded area). (c) The resulting trajectory distribution produced by the inverse covariance control approach (blue). Due to latency-effects it missed the via-points in time and generated high actions which led to the velocity spike. (d) Trajectory distribution produced by DMPs. While the DMP can follow the mean of the demonstrations, it can not adapt its variance. The accuracy at the via-points is worse than ProMPs, while the control actions are higher in non-relevant areas of the trajectory. In blue we tuned the DMP gains for reproducing the trajectory distribution with the lowest cost and in green we used lower gains.

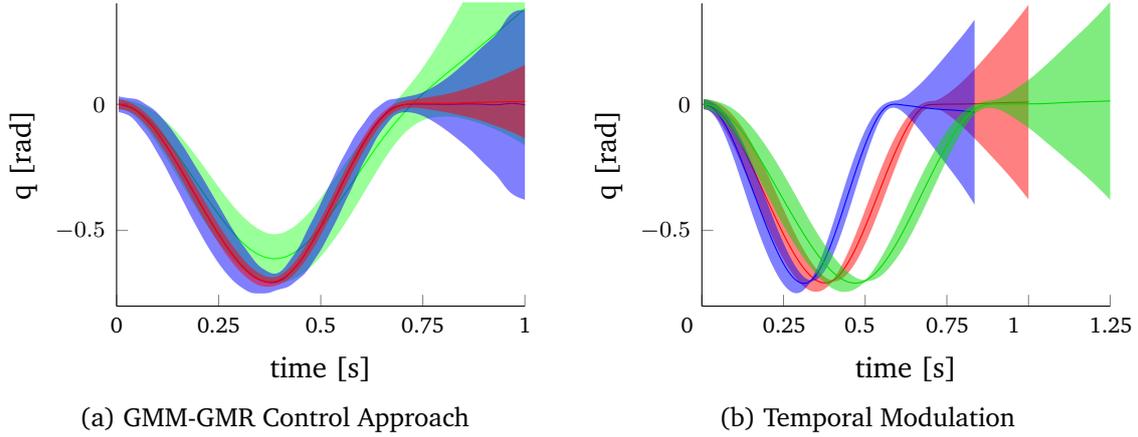


Figure 2.3: (a) Evaluation of the GMM-GMR approach, using the minimum intervention principle for control. The learned distribution using the GMM-GMR approach is presented in blue. The approach captures the mean of the distribution accurately, however, the variance at the via-points is higher than in the demonstrations. For reproduction, we used the optimal action penalty (red) or the same action penalty as in the demonstrations (green). While the mean of the reproductions matches the mean of the demonstrations, there is a mismatch for the variance. (b) Temporal Modulation of the ProMPs. The demonstrated distribution is shown in red. The green shows an execution at a slower pace, whereas the blue at a faster one.

the speed of the movement. Without loss of generality, we define the phase as $z_0 = 0$ at the beginning of the movement and as $z_T = 1$ at the end. We typically use a constant velocity $\dot{z}_t = 1/T$ for reproducing the recorded motion, but we can also adapt it dynamically during the execution of the movement. The basis functions ϕ_t now directly depend on the phase instead of time, such that

$$\phi_t = \phi(z_t), \quad \dot{\phi}_t = \dot{\phi}(z_t)\dot{z}_t, \quad (2.5)$$

where $\dot{\phi}_t$ denotes the corresponding derivative. An illustration of temporal scaling for our running example is shown in Figure 2.3b.

Rhythmic and Stroke-Based Movements

The choice of the basis functions depends on the type of movement, which can be either rhythmic or stroke-based. For stroke-based movements, we use Gaussian basis functions b_i^G , while for rhythmic movements, we use Von-Mises basis functions b_i^{VM} to model periodicity in the phase variable z , i.e.,

$$b_i^G(z) = \exp\left(-\frac{(z_t - c_i)^2}{2h}\right), \quad b_i^{VM}(z) = \exp\left(\frac{\cos(2\pi(z_t - c_i))}{h}\right), \quad (2.6)$$

where h defines the width of the basis and c_i the center for the i th basis function. We normalize the basis functions

$$\phi_i(z_t) = \frac{b_i(z)}{\sum_{j=1}^n b_j(z)}, \quad (2.7)$$

to obtain a constant summed activation and improve the regression’s performance. The centers of the basis functions are uniformly placed in $[-2h, (1 + 2h)]$ the phase domain. We center basis functions outside the interval $[0, 1]$ to improve homogeneity of the basis vector, i.e., by including the “tails” of the basis placed outside, and therefore improve the performance of our model.

Encoding Coupling between Joints

So far, we have considered each degree of freedom to be modeled independently. However, for many tasks we have to coordinate the movement of multiple joints. The trajectory distributions $p(\boldsymbol{\tau}; \boldsymbol{\theta})$ can be easily extended to the multi-DoF case. For each dimension i , we maintain a parameter vector \mathbf{w}_i , and we define the combined weight vector \mathbf{w} as $\mathbf{w} = [\mathbf{w}_1^T, \dots, \mathbf{w}_n^T]^T$, a concatenation of the weight vectors. The basis matrix Φ_t now extends to a block-diagonal matrix containing the basis functions and their derivatives for each dimension. The observation vector \mathbf{y}_t consists of the angles and velocities of all joints. The probability of an observation \mathbf{y} at time t is given by

$$p(\mathbf{y}_t | \mathbf{w}) = \mathcal{N} \left(\begin{bmatrix} \mathbf{y}_{1,t} \\ \vdots \\ \mathbf{y}_{d,t} \end{bmatrix} \middle| \begin{bmatrix} \Phi_t & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \Phi_t \end{bmatrix} \mathbf{w}, \Sigma_y \right) = \mathcal{N}(\mathbf{y}_t | \Psi_t \mathbf{w}, \Sigma_y) \quad (2.8)$$

where $\mathbf{y}_{i,t} = [q_{i,t}, \dot{q}_{i,t}]^T$ denotes the joint angle and velocity for the i^{th} joint. We now maintain a distribution $p(\mathbf{w}; \boldsymbol{\theta})$ over the combined parameter vector \mathbf{w} . By introducing $p(\mathbf{w}; \boldsymbol{\theta})$, we extended our representation to additionally capture the correlation between the joints. The extended multi-DoF representation is used throughout the rest of the chapter, including the experimental section. Controlling the robot in a co-ordinated manner using the coupling between the joints, for example, allows the robot to reach a via-point defined in the task-space while the joints exhibit variability. In the multi-DoF model, Equation (2.2) becomes

$$p(\boldsymbol{\tau} | \mathbf{w}) = \prod_t \mathcal{N}(\mathbf{y}_t | \Psi_t \mathbf{w}, \Sigma_y). \quad (2.9)$$

Additionally, our model captures the covariance of joint positions and velocities for each time step. Therefore, it encodes a linear relationship between them and enables to compute the desired velocity if the position is known or vice versa. We further exploit this property in Sec. 2.4.3 for adaptation to novel situations.

Algorithm 1: Learning Stroke-Based Movements

Data: A set of N trajectories with position observations $\mathbf{Y}_i, i = 1 \dots N$.

Input: Number of basis functions K , Basis function width h , Regression parameter λ .

Result: The mean $\boldsymbol{\mu}_w$ and covariance $\boldsymbol{\Sigma}_w$ of $p(\mathbf{w}) \sim \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w)$.

foreach trajectory i do

→ Compute phase: $\mathbf{z}_i = t_i/t_i^{\text{end}}$;

→ Generate basis: $\boldsymbol{\Psi} = f(\mathbf{z}_i, K, b)$, Equation (2.7);

→ Compute the weight vector \mathbf{w}_i for trajectory i

$$\mathbf{w}_i = (\boldsymbol{\Psi}^T \boldsymbol{\Psi} + \lambda \mathbf{I})^{-1} \boldsymbol{\Psi}^T \mathbf{Y}_i.$$

end

→ Fit a Gaussian over the weight vectors \mathbf{w}_i

$$\boldsymbol{\mu}_w = \frac{1}{N} \sum_{i=1}^N \mathbf{w}_i, \quad \boldsymbol{\Sigma}_w = \frac{1}{N-1} \sum_{i=1}^N (\mathbf{w}_i - \boldsymbol{\mu}_w)(\mathbf{w}_i - \boldsymbol{\mu}_w)^T.$$

return $\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w$.

2.4.2 Learning from Demonstrations

To simplify the learning of the parameters $\boldsymbol{\theta}$, we will assume a Gaussian distribution for $p(\mathbf{w}; \boldsymbol{\theta}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w)$ over the parameters \mathbf{w} . Consequently, the distribution of the state $p(\mathbf{y}_t|\boldsymbol{\theta})$ for time step t is given by

$$p(\mathbf{y}_t; \boldsymbol{\theta}) = \int \mathcal{N}(\mathbf{y}_t|\boldsymbol{\Psi}_t \mathbf{w}, \boldsymbol{\Sigma}_y) \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w) d\mathbf{w} = \mathcal{N}(\mathbf{y}_t|\boldsymbol{\Psi}_t \boldsymbol{\mu}_w, \boldsymbol{\Psi}_t \boldsymbol{\Sigma}_w \boldsymbol{\Psi}_t^T + \boldsymbol{\Sigma}_y), \quad (2.10)$$

and, thus, we can easily evaluate the mean and the variance for any time point t . As a ProMP represents multiple ways to execute an elemental movement, we need multiple demonstrations in order to learn $p(\mathbf{w}; \boldsymbol{\theta})$, or, in the special case that only one demonstration is available, a prior variance profile for $p(\mathbf{w})$ should be given².

Learning Stroke-based Movements

For stroke-based movements, we can estimate the parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w\}$ from demonstrations by a simple maximum likelihood estimation algorithm. We estimate the weights for each trajectory individually with linear ridge regression, i.e.

$$\mathbf{w}_i = (\boldsymbol{\Psi}^T \boldsymbol{\Psi} + \lambda \mathbf{I})^{-1} \boldsymbol{\Psi}^T \mathbf{Y}_i \quad (2.11)$$

² This prior variance profile can be just set to $\alpha \mathbf{I}$, where α is a small constant and \mathbf{I} is the identity matrix.

where \mathbf{Y}_i represents the positions of all joints and time steps from the demonstration i , and Ψ the corresponding basis function matrix for all time steps. We align the demonstrations by adjusting the phase signal. For each demonstration, we assume that $z_{\text{begin}} = 0$ and at the end $z_{\text{end}} = 1$. The ridge factor λ is generally set to a very small value, typically $\lambda = 10^{-12}$, as larger values degrade the estimation the trajectory distribution. In this chapter, we also propose a new regularization scheme that is based on minimizing the jerk of the trajectories, i.e.,

$$\mathbf{w}_i = (\Psi\Psi + \lambda\Gamma^T\Gamma)^{-1} \Psi^T \mathbf{Y}_i, \quad (2.12)$$

where Γ denotes the third derivative³ of Ψ . The third derivative is needed as the jerk is given by the third derivative. The jerk minimization scheme can generate smoother torque profiles and, hence, performs better in the cost function comparison presented in Table 2.2. The mean μ_w and covariance Σ_w are computed from the samples \mathbf{w}_i ,

$$\mu_w = \frac{1}{N} \sum_{i=1}^N \mathbf{w}_i, \quad \hat{\Sigma}_w = \frac{1}{N-1} \sum_{i=1}^N (\mathbf{w}_i - \mu_w)(\mathbf{w}_i - \mu_w)^T, \quad (2.13)$$

where N is the number of demonstrations. We use an Inverse-Wishart distribution as a prior to the covariance matrix Σ_w . The maximum a-posteriori estimate of the covariance (O’Hagan and Forster, 2004) given the prior becomes

$$\Sigma_w = \frac{N\hat{\Sigma}_w + \lambda_w \mathbf{I}}{N + \lambda}, \quad (2.14)$$

where the value of λ_w is set such that the covariance matrix Σ_w is positive-definite. The complete algorithm is shown in Algorithm 1.

Learning Periodic Movements

In this section we present an Expectation-Maximization (EM) algorithm that can be used to learn from missing data or rhythmic movements. Using the previous learning approach for periodic movements would require that each demonstration finishes at the same state as it started, as we use a single weight vector per demonstration and the basis functions are periodic. However, due to the variability, single trajectories typically do not end exactly where they started. Yet, rhythmic movements can be learned by using an EM-algorithm that we can train with partial trajectories, i.e., trajectories that do not cover a whole period.

We derive an Expectation Maximization (EM) algorithm that infers the latent variables, i.e. the weights for each demonstrations during training (Ewerton et al., 2015a). We assume that our set of demonstrations contains multiple periods. First, we determine the period length from the demonstration and we construct the basis and phase signal. We randomly split the demonstration to N potentially overlapping segments. The size of the segment must be shorter than a period to avoid the pe-

³ The third derivative of Ψ can be computed numerically.

riodicity in the basis functions for a single demonstration. The initial guess for the parameters is estimated using linear ridge regression. In the expectation step, we need to compute the posterior distribution of the weights

$$p(\mathbf{w}_i | \mathbf{Y}_i, \boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w) \propto p(\mathbf{Y}_i | \mathbf{w}_i) p(\mathbf{w}_i | \boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w), \quad (2.15)$$

for each demonstration. The posterior can be computed using the Bayes rule for Gaussian distributions. The expectation step becomes

$$\boldsymbol{\mu}_i = \boldsymbol{\mu}_w + \boldsymbol{\Psi}_i^T (\boldsymbol{\Psi}_i \boldsymbol{\Sigma}_w \boldsymbol{\Psi}_i^T)^{-1} (\mathbf{Y}_i - \boldsymbol{\Psi}_i \boldsymbol{\mu}_w), \quad (2.16)$$

$$\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}_w - \boldsymbol{\Sigma}_w \boldsymbol{\Psi}_i^T (\boldsymbol{\Psi}_i \boldsymbol{\Sigma}_w \boldsymbol{\Psi}_i^T)^{-1} \boldsymbol{\Psi}_i \boldsymbol{\Sigma}_w, \quad (2.17)$$

where the index i denotes the i -th segment of the demonstration and $\boldsymbol{\Psi}_i$ the basis functions for that segment. We dropped the time dependency from the notation of $\boldsymbol{\Psi}_i$ for clearness. In the maximization step, we need to optimize the complete-data log-likelihood

$$\operatorname{argmax}_{\boldsymbol{\theta}'} \sum_{i=1}^N \int_{\mathbf{w}} p(\mathbf{w}_i | \boldsymbol{\theta}') \log p(\mathbf{Y}_i | \boldsymbol{\theta}') p(\mathbf{w} | \boldsymbol{\theta}') d\mathbf{w} \quad (2.18)$$

where $\boldsymbol{\theta}' = \{\boldsymbol{\mu}'_w, \boldsymbol{\Sigma}'_w\}$ denote the new parameters for the weight distribution. Thus, the maximization step becomes

$$\boldsymbol{\mu}'_w = \frac{1}{N} \sum_{i=1}^N \boldsymbol{\mu}_i, \quad \boldsymbol{\Sigma}'_w = \frac{1}{N} \sum_{i=1}^N \left((\boldsymbol{\mu}_i - \boldsymbol{\mu}'_w) (\boldsymbol{\mu}_i - \boldsymbol{\mu}'_w)^T + \boldsymbol{\Sigma}_i \right), \quad (2.19)$$

for computing the updates in closed form. We iterate between the expectation step and the maximization step until convergence. Our algorithm is based on the EM from HBMs with Gaussian distributions approach presented in (Lazarcic and Ghavamzadeh, 2010) and has been evaluated in (Paraschos et al., 2013a; Ewerton et al., 2015a) for the ProMP representation. The algorithm for learning periodic movements is shown in Algorithm 2.

In both learning approaches, the weight covariance $\boldsymbol{\Sigma}_w$ may become not positive definite because of numerical problems. To correct these numerical problems we use an eigen-decomposition to find the closest symmetric positive definite matrix to our estimation, as described in (Higham, 1988).

2.4.3 New Probabilistic Operators for Movement Primitives

With the probabilistic representation we can exploit probabilistic operators, i.e., modulate the trajectory by conditioning and co-activate MPs by computing the product of distributions. Using Gaussian distributions for $p(\mathbf{w}; \boldsymbol{\theta})$, all operators can be computed in closed form.

Algorithm 2: Learning Periodic Movements

Data: A trajectory with multiple periods with position observations \mathbf{Y} , at time t

Input: Number of basis functions K , Basis function width b , Regression parameter λ , Number of segments to split N , EM convergence parameter ϵ

Result: The mean $\boldsymbol{\mu}_w$ and covariance $\boldsymbol{\Sigma}_w$ of $p(\mathbf{w}) \sim \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w)$

→ Detect base frequency: f_q by FFT;

→ Periodic phase signal: $\mathbf{z} = \text{mod}(tf_q, 1)$;

→ Split randomly: $\{\mathbf{Y}, \mathbf{z}\}$ into N segments;

→ Initial guess: $\boldsymbol{\mu}_w$ and $\boldsymbol{\Sigma}_w$ from Algorithm 1;

repeat

 Expectation step:

$$\begin{aligned}\boldsymbol{\mu}_i &= \boldsymbol{\mu}_w + \boldsymbol{\Psi}_i^T (\boldsymbol{\Psi}_i \boldsymbol{\Sigma}_w \boldsymbol{\Psi}_i^T)^{-1} (\mathbf{Y}_i - \boldsymbol{\Psi}_i \boldsymbol{\mu}_w), \\ \boldsymbol{\Sigma}_i &= \boldsymbol{\Sigma}_w - \boldsymbol{\Sigma}_w \boldsymbol{\Psi}_i^T (\boldsymbol{\Psi}_i \boldsymbol{\Sigma}_w \boldsymbol{\Psi}_i^T)^{-1} \boldsymbol{\Psi}_i \boldsymbol{\Sigma}_w\end{aligned}$$

 Maximization step:

$$\boldsymbol{\mu}'_w = \frac{1}{N} \sum_{i=1}^N \boldsymbol{\mu}_i, \quad \boldsymbol{\Sigma}'_w = \frac{1}{N} \sum_{i=1}^N \left((\boldsymbol{\mu}_i - \boldsymbol{\mu}'_w) (\boldsymbol{\mu}_i - \boldsymbol{\mu}'_w)^T + \boldsymbol{\Sigma}_i \right)$$

until difference in log-likelihood $< \epsilon$;

return $\boldsymbol{\mu}'_w, \boldsymbol{\Sigma}'_w$.

Modulation of the Trajectory Distribution by Conditioning

The modulation of via-points and final positions is an important property to adapt the MP to new situations. In our probabilistic formulation, such operations can be described by conditioning the MP to reach a certain state \mathbf{y}_t^* at time t . Note that conditioning can be performed for any time point t . It is performed by adding a desired observation $\mathbf{x}_t^* = \{\mathbf{y}_t^*, \boldsymbol{\Sigma}_y^*\}$ to our probabilistic model and applying Bayes theorem

$$p(\mathbf{w}|\mathbf{x}_t^*) \propto \mathcal{N}(\mathbf{y}_t^*|\boldsymbol{\Psi}_t \mathbf{w}, \boldsymbol{\Sigma}_y^*) p(\mathbf{w}), \quad (2.20)$$

where the state vector \mathbf{y}_t^* represents the desired position and velocity vector at time t and $\boldsymbol{\Sigma}_y^*$ describes the accuracy of the desired observation. We can also condition on any subset of \mathbf{y}_t^* . For example, specifying a desired joint position q_1 for the first joint the trajectory distribution will automatically infer the most probable joint positions for the other joints. Conditioning partially on the state is done by constructing the basis function matrix $\boldsymbol{\Psi}$ used in Equation (2.21) to contain only the variables that participate in the conditioning. For example, Maeda et al. (Maeda et al., 2014) used such an approach based on ProMPs to model human-robot interaction where conditioning on the human movement yields the desired movement of the robot.

For Gaussian trajectory distributions, the conditional distribution $p(\mathbf{w}|\mathbf{x}_t^*)$ for \mathbf{w} is Gaussian with mean and variance

$$\boldsymbol{\mu}_w^{[\text{new}]} = \boldsymbol{\mu}_w + \mathbf{L}(\mathbf{y}_t^* - \boldsymbol{\Psi}_t^T \boldsymbol{\mu}_w), \quad \boldsymbol{\Sigma}_w^{[\text{new}]} = \boldsymbol{\Sigma}_w - \mathbf{L} \boldsymbol{\Psi}_t^T \boldsymbol{\Sigma}_w, \quad (2.21)$$

where \mathbf{L} is

$$\mathbf{L} = \boldsymbol{\Sigma}_w \boldsymbol{\Psi}_t (\boldsymbol{\Sigma}_y^* + \boldsymbol{\Psi}_t^T \boldsymbol{\Sigma}_w \boldsymbol{\Psi}_t)^{-1}. \quad (2.22)$$

Illustrative Example. Conditioning a ProMP to different target states, positions and velocities, is illustrated in Figure 2.4. We observe that, despite the modulation of the ProMP by conditioning, the ProMP stays within the original distribution. How the ProMPs modulate is hence learned from the original demonstrations. Modulation strategies in other approaches such as the DMPs do not show this effect (Schaal et al., 2003b). DMPs can reach the desired target position and velocities at the end of the movement, but deform the trajectory significantly. In contrast, the trajectory distribution obtained by conditioning a ProMP even matches the distribution of the optimal controller that has the conditioned via-point as additional cost term.

Adaptation to Task Parameters

In many situations, we need to adapt the primitive based on an external state variable $\hat{\mathbf{s}}$, such as a desired target angle when shooting hockey pucks. The value of such external variables is typically known during training and also before reproduction of the primitive. Hence, we can directly learn this adaptation by learning a mapping from the external variable to the mean weight vector $\boldsymbol{\mu}_w$. We use a simple linear mapping, which is equivalent to modeling a joint distribution

$$p(\mathbf{w}, \hat{\mathbf{s}}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{w} \\ \hat{\mathbf{s}} \end{bmatrix} \middle| \boldsymbol{\mu}, \boldsymbol{\Sigma}\right) = \mathcal{N}(\mathbf{w} | \mathbf{O} \hat{\mathbf{s}} + \mathbf{o}, \boldsymbol{\Sigma}_w) \mathcal{N}(\hat{\mathbf{s}} | \boldsymbol{\mu}_{\hat{\mathbf{s}}}, \boldsymbol{\Sigma}_{\hat{\mathbf{s}}}), \quad (2.23)$$

however, the transformation parameters $\{\mathbf{O}, \mathbf{o}\}$ are learned directly with linear ridge regression.

Combination and Blending of Movement Primitives

We can use a product of trajectory distributions to continuously combine and blend different MPs into a single movement. Suppose that we maintain a set of i different primitives that we want to combine. We can co-activate them by taking the products of distributions,

$$p_{\text{new}}(\boldsymbol{\tau}) \propto \prod_i p_i(\boldsymbol{\tau})^{\alpha^{[i]}}, \quad (2.24)$$

where the $\alpha^{[i]} \in [0, 1]$ factors denote the activation of the i^{th} primitive. The product captures the overlapping region of the active MPs, i.e., the part of the trajectory space where all MPs have high probability mass.

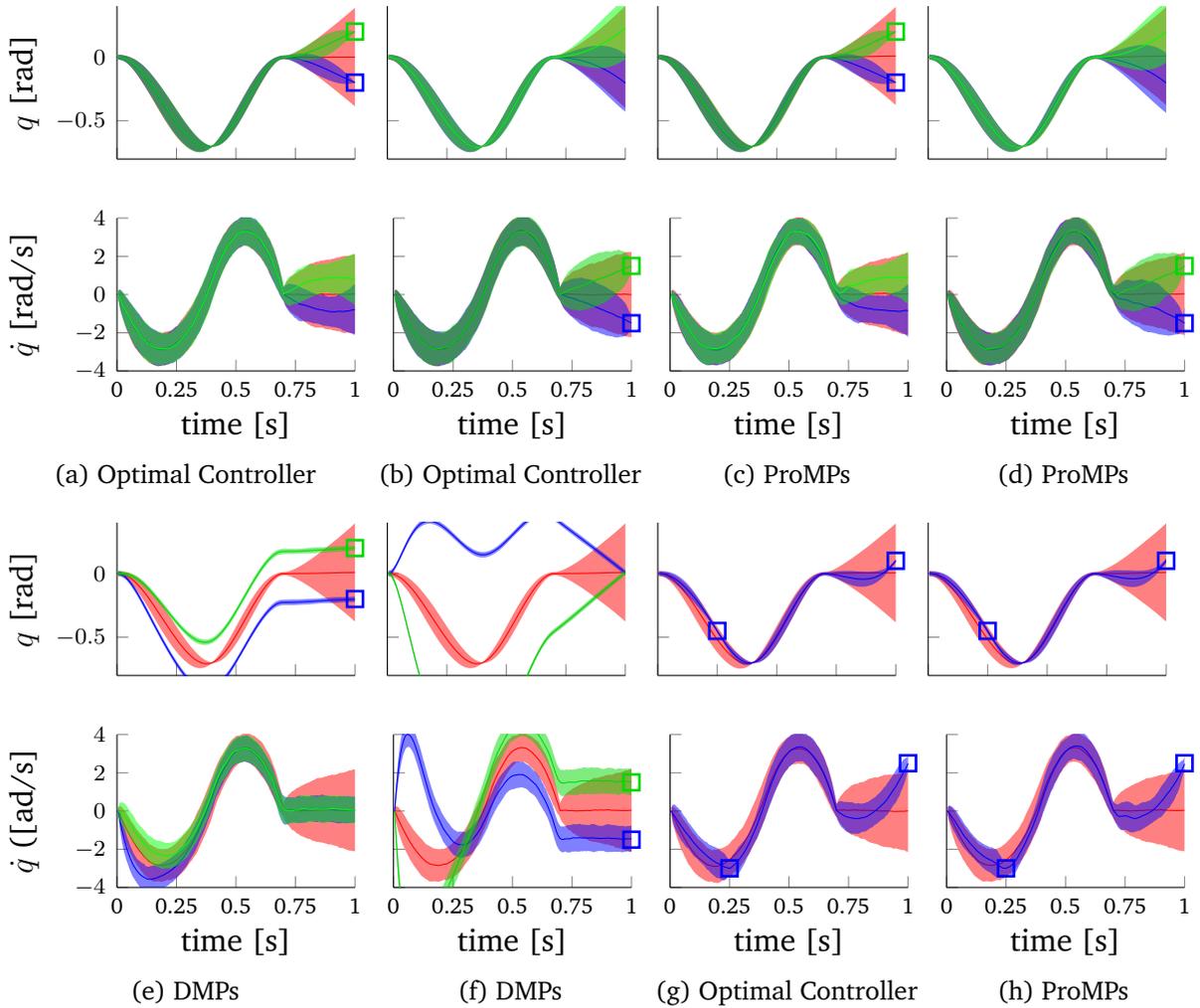


Figure 2.4: *Generalization of primitives.* We want to modulate the MPs such that they go through additional via-points (blue and green) and evaluate the quality of the generalized MP policies. The resulting distributions are illustrated only for comparison and are not used for training. The added via-points are depicted with colored boxes. (a,b) Evaluation of the optimal controller given the additional via-points on the final position (a) or final velocity (b). (c,d) Evaluation of the ProMP on the same via-points. ProMPs reproduce the optimal behavior despite that the unconditioned demonstrations have been used for training. (e,f) Generalization to the same via-points with DMPs. The position generalization is a linear interpolation of the mean trajectory and quickly goes “outside” the demonstrated distribution. The final velocity generalization reproduce drastically different trajectories than the demonstrated ones. (g,h) Evaluation of the optimal controller and the ProMPs on additional via-point in intermediate and final locations, that require adaptation on both the position and the velocity simultaneously.

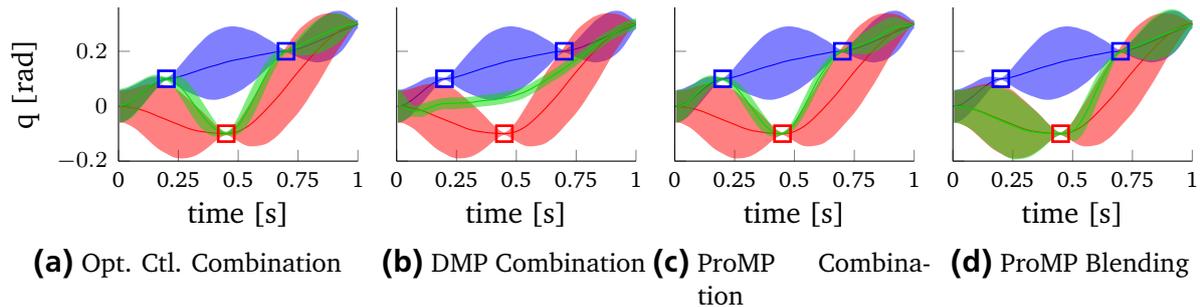


Figure 2.5: *Combination and blending of two primitives.* We want to combine two MPs to obtain an MP that can achieve both tasks of the single MPs at the same time. We show the resulting distribution in green and the participating primitives in blue and red. (a) The resulting optimal distribution is generated by adding both cost-functions that have been used to generate the single primitive distributions. (b) Combining DMPs linearly in weight space results in a linearly interpolated trajectory. The movement misses all the via-points. (c) We co-activate two ProMPs with equal weights. The resulting movement passes through all via-points. (d) We smoothly blend from the red primitive to the blue primitive. The resulting movement (green) first follows the red primitive and, subsequently, switches to following exactly the blue primitive.

We also want to be able to modulate the activations of the primitives, for example, to continuously blend the movement execution from one primitive to the next one. Hence, we decompose the trajectory into its single time steps and use time-varying activation functions $\alpha_t^{[i]}$, i.e.,

$$p^*(\boldsymbol{\tau}) \propto \prod_t \prod_i p_i(\mathbf{y}_t)^{\alpha_t^{[i]}}, \quad p_i(\mathbf{y}_t) = \int p_i(\mathbf{y}_t | \mathbf{w}^{[i]}) p_i(\mathbf{w}^{[i]}) d\mathbf{w}^{[i]}. \quad (2.25)$$

For Gaussian distributions $p_i(\mathbf{y}_t) = \mathcal{N}(\mathbf{y}_t | \boldsymbol{\mu}_t^{[i]}, \boldsymbol{\Sigma}_t^{[i]})$, the resulting distribution $p^*(\mathbf{y}_t)$ is again Gaussian with variance and mean,

$$\boldsymbol{\Sigma}_t^* = \left(\sum_i \left(\boldsymbol{\Sigma}_t^{[i]} / \alpha_t^{[i]} \right)^{-1} \right)^{-1}, \quad \boldsymbol{\mu}_t^* = \boldsymbol{\Sigma}_t^* \left(\sum_i \left(\boldsymbol{\Sigma}_t^{[i]} / \alpha_t^{[i]} \right)^{-1} \boldsymbol{\mu}_t^{[i]} \right). \quad (2.26)$$

Illustrative Example. Co-activation of two ProMPs is shown in Figure 2.5(c) and blending of two ProMPs in Figure 2.5(d). We trained the ProMPs such that each primitive solves a different task indicated by the via points in the figures with the same colors. The combined primitive is capable of reaching all four via-points, i.e., it achieved both tasks at the *same* time. Additionally, we compare our combination approach to the optimal controller by adding the cost functions of the two tasks. The optimal controller results are shown in Figure 2.5(a). Combining movements with the DMPs results on averaging between the trajectories and therefore missing all of

the via-points. The trajectory distribution is shown in Figure 2.5(b). We quantified the results in terms of the average cost in Table 2.2. While the ProMP approach achieves an average cost in the same range of magnitude, the performance of the DMP combination is highly degraded.

2.4.4 Using Trajectory Distributions for Robot Control

In order to fully exploit the properties of trajectory distributions, a policy that reproduces these distributions is needed for controlling the robot. To this effect, we derive a stochastic feedback controller that can accurately reproduce the mean $\boldsymbol{\mu}_t$, the variances $\boldsymbol{\Sigma}_t$, and the correlations $\boldsymbol{\Sigma}_{t,t+1}$ for all time steps t of a given trajectory distribution. The derivation of the controller is based on moment matching on Gaussian distribution. In our approach there is no notion of cost function.

Such controller can only be obtained by using a model. We approximate the continuous time dynamics of the system by a linearized discrete-time system with step duration Δt ,

$$\mathbf{y}_{t+\Delta t} = (\mathbf{I} + \mathbf{A}_t \Delta t) \mathbf{y}_t + \mathbf{B}_t \Delta t \mathbf{u} + \mathbf{c}_t \Delta t, \quad (2.27)$$

where the system matrices \mathbf{A}_t , the input matrices \mathbf{B}_t and the drift vectors \mathbf{c}_t can be obtained by first order Taylor expansion of the dynamical system for the current state \mathbf{y}_t ⁴. We assume a stochastic linear feedback controller with time varying feedback gains is generating the control actions, i.e.,

$$\mathbf{u} = \mathbf{K}_t \mathbf{y}_t + \mathbf{k}_t + \boldsymbol{\epsilon}_u, \quad \boldsymbol{\epsilon}_u \sim \mathcal{N}(\boldsymbol{\epsilon}_u | 0, \boldsymbol{\Sigma}_u \Delta t^{-1}), \quad (2.28)$$

where the matrix \mathbf{K}_t denotes a feedback gain matrix and \mathbf{k}_t a feed-forward component. We use a control noise which behaves like a Wiener process (Stark and Woods, 2001), and, hence, its variance grows linearly with the step duration⁵ Δt . By substituting Equation (2.28) into Equation (2.27), we can rewrite the next state of the system as

$$\begin{aligned} \mathbf{y}_{t+\Delta t} &= (\mathbf{I} + (\mathbf{A}_t + \mathbf{B}_t \mathbf{K}_t) \Delta t) \mathbf{y}_t + \mathbf{B}_t \Delta t (\mathbf{k}_t + \boldsymbol{\epsilon}_u) + \mathbf{c}_t \Delta t \\ &= \mathbf{F}_t \mathbf{y}_t + \mathbf{f}_t + \mathbf{B}_t \Delta t \boldsymbol{\epsilon}_u, \end{aligned} \quad (2.29)$$

where we defined

$$\mathbf{F}_t = (\mathbf{I} + (\mathbf{A}_t + \mathbf{B}_t \mathbf{K}_t) \Delta t), \quad \mathbf{f}_t = \mathbf{B}_t \mathbf{k}_t \Delta t + \mathbf{c}_t \Delta t. \quad (2.30)$$

We will omit the time-index as subscript for most matrices in the remainder of the chapter to improve readability. From Equation (2.10), we know that the distribution

⁴ If inverse dynamics control (Peters et al., 2008) is used for the robot, the system reduces to a linear system where the terms \mathbf{A}_t , \mathbf{B}_t and \mathbf{c}_t are constant in time.

⁵ As we multiply the noise by $\mathbf{B} \Delta t$, we need to divide the covariance $\boldsymbol{\Sigma}_u$ of the control noise $\boldsymbol{\epsilon}_u$ by Δt to obtain this desired behavior.

for our current state \mathbf{y}_t is Gaussian with mean $\boldsymbol{\mu}_t = \boldsymbol{\Psi}_t \boldsymbol{\mu}_w$ and covariance⁶ $\boldsymbol{\Sigma}_t = \boldsymbol{\Psi}_t \boldsymbol{\Sigma}_w \boldsymbol{\Psi}_t^T$. As the system dynamics are modeled by a Gaussian linear model, we can obtain the distribution of the next state $p(\mathbf{y}_{t+\Delta t})$ analytically from the forward model by integrating out the current state

$$\begin{aligned} p(\mathbf{y}_{t+\Delta t}) &= \int_{\mathbf{y}_t} \mathcal{N}(\mathbf{y}_{t+\Delta t} | \mathbf{F}\mathbf{y}_t + \mathbf{f}, \boldsymbol{\Sigma}_s \Delta t) \mathcal{N}(\mathbf{y}_t | \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) \\ &= \mathcal{N}(\mathbf{y}_{t+\Delta t} | \mathbf{F}\boldsymbol{\mu}_t + \mathbf{f}, \mathbf{F}\boldsymbol{\Sigma}_t \mathbf{F}^T + \boldsymbol{\Sigma}_s \Delta t), \end{aligned} \quad (2.31)$$

where $\Delta t \boldsymbol{\Sigma}_s = \Delta t \mathbf{B} \boldsymbol{\Sigma}_u \mathbf{B}^T$ represents the system noise matrix. Both sides of Equation (2.31) are Gaussian distributions. The left-hand side can be computed in two ways; from our desired trajectory distribution $p(\boldsymbol{\tau}; \boldsymbol{\theta})$ and from Equation (2.31). We proceed by matching the mean and the variances of both sides with our control law,

$$\boldsymbol{\mu}_{t+\Delta t} = \mathbf{F}\boldsymbol{\mu}_t + (\mathbf{B}\mathbf{k} + \mathbf{c})\Delta t, \quad \boldsymbol{\Sigma}_{t+\Delta t} = \mathbf{F}\boldsymbol{\Sigma}_t \mathbf{F}^T + \boldsymbol{\Sigma}_s \Delta t, \quad (2.32)$$

where \mathbf{F} is given in Equation (2.30) and contains the time varying feedback gains \mathbf{K} . Using both constraints, we can now obtain the time-dependent gains \mathbf{K}_t and \mathbf{k}_t . Note that the linearized model given by \mathbf{A}_t , \mathbf{B}_t and \mathbf{c}_t depends on the current state \mathbf{y}_t which is used as linearization point. As our computation of the gains will depend on the linearized model, our controller gains also depend implicitly on the current state, i.e., $\mathbf{K}_t = \mathbf{K}(\mathbf{y}_t)$ and $\mathbf{k}_t = \mathbf{k}(\mathbf{y}_t)$. Therefore, our controller is in fact a non-linear controller. However, we will omit the state dependence of our gains in the remaining derivation for the sake of clarity.

Derivation of the Controller Gains

We continue with the derivation of the controller gains, \mathbf{K} . To perform the derivation we assume, for the moment, that the stochasticity of the controller $\boldsymbol{\Sigma}_u$ is known. Further, we show how the stochasticity of the controller can be computed in closed form. By rearranging terms, the covariance constraint becomes

$$\boldsymbol{\Sigma}_{t+\Delta t} - \boldsymbol{\Sigma}_t = \boldsymbol{\Sigma}_s \Delta t + (\mathbf{A} + \mathbf{B}\mathbf{K}) \boldsymbol{\Sigma}_t \Delta t + \boldsymbol{\Sigma}_t (\mathbf{A} + \mathbf{B}\mathbf{K})^T \Delta t + O(\Delta t^2), \quad (2.33)$$

where $O(\Delta t^2)$ denotes all second order terms in Δt . After dividing by Δt and taking the limit of $\Delta t \rightarrow 0$, the second order terms disappear and we obtain the time derivative of the covariance

$$\dot{\boldsymbol{\Sigma}}_t = \lim_{\Delta t \rightarrow 0} \frac{\boldsymbol{\Sigma}_{t+\Delta t} - \boldsymbol{\Sigma}_t}{\Delta t} = (\mathbf{A} + \mathbf{B}\mathbf{K}) \boldsymbol{\Sigma}_t + \boldsymbol{\Sigma}_t (\mathbf{A} + \mathbf{B}\mathbf{K})^T + \boldsymbol{\Sigma}_s, \quad (2.34)$$

which is a special case of the continuous time Riccati equation. Note that this operation was only possible due to the continuous time formulation of the basis functions.

⁶ The observation noise is omitted as it represents independent noise which is not used for predicting the next state.

The derivative of the covariance matrix $\dot{\Sigma}_t$ can also be obtained from the trajectory distribution by

$$\dot{\Sigma}_t = \dot{\Psi}_t \Sigma_w \Psi_t^T + \Psi_t \Sigma_w \dot{\Psi}_t^T, \quad (2.35)$$

which we substitute into Equation (2.34). After rearranging terms, the equation reads

$$M + M^T = BK\Sigma_t + (BK\Sigma_t)^T, \quad (2.36)$$

where we defined

$$M = \dot{\Psi}_t \Sigma_w \Psi_t^T - A\Sigma_t - 0.5\Sigma_s, \quad (2.37)$$

to demonstrate the structure of the equation. A solution can be obtained by setting $M = BK\Sigma_t$ and solving for the gain matrix K ,

$$K = B^\dagger \left(\dot{\Psi}_t \Sigma_w \Psi_t^T - A\Sigma_t - 0.5\Sigma_s \right) \Sigma_t^{-1}, \quad (2.38)$$

where B^\dagger denotes the pseudo-inverse of the control matrix B .

Derivation of the Feed-Forward Controls

Similarly, we obtain the feed-forward control signal k by matching the mean of the trajectory distribution $\mu_{t+\Delta t}$ with the mean computed with the forward model. After rearranging terms, dividing by Δt , and taking the limit of $\Delta t \rightarrow 0$, we arrive at

$$\dot{\mu}_t = (A + BK) \mu_t + Bk + c, \quad (2.39)$$

the differential equation for the mean of the trajectory. We use the trajectory distribution $p(\tau; \theta)$ to obtain $\mu_t = \Psi_t \mu_w$ and $\dot{\mu}_t = \dot{\Psi}_t \mu_w$ and solve Equation (2.39) for k ,

$$k = B^\dagger \left(\dot{\Psi}_t \mu_w - (A + BK) \Psi_t \mu_w - c \right). \quad (2.40)$$

The time-varying feedback gains K do not depend on the mean of the trajectory distribution, but only on the variance at that time step. Similarly, the feed-forward controls k , depend on the variance only through the feedback gains K , but otherwise they depend on the mean.

Estimation of the Control Noise

The last step required to match the trajectory distribution is to match the control noise matrix Σ_u which is needed to generate the distribution. This noise can be higher than the system noise to induce a higher variance in the distribution. Such a higher variance can, for example, be useful for exploration in reinforcement learning.

We compute the system noise covariance $\Sigma_s = B\Sigma_u B^T$ by examining the cross-correlation between time steps of the trajectory distribution. To do so, we compute the joint distribution $p(\mathbf{y}_t, \mathbf{y}_{t+\Delta t})$ of the current state \mathbf{y}_t and the next state $\mathbf{y}_{t+\Delta t}$ as

$$p(\mathbf{y}_t, \mathbf{y}_{t+\Delta t}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{y}_t \\ \mathbf{y}_{t+\Delta t} \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\mu}_t \\ \boldsymbol{\mu}_{t+\Delta t} \end{bmatrix}, \begin{bmatrix} \Sigma_t & \mathbf{C}_t \\ \mathbf{C}_t^T & \Sigma_{t+\Delta t} \end{bmatrix}\right), \quad (2.41)$$

where $\mathbf{C}_t = \Psi_t \Sigma_w \Psi_{t+\Delta t}^T$ is the cross-correlation of the subsequent time points. We use our linear Gaussian model to match the cross correlation. The joint distribution for \mathbf{y}_t and $\mathbf{y}_{t+\Delta t}$ can also be obtained by our system dynamics, i.e.,

$$p(\mathbf{y}_t, \mathbf{y}_{t+\Delta t}) = \mathcal{N}(\mathbf{y}_t | \boldsymbol{\mu}_t, \Sigma_t) \mathcal{N}(\mathbf{y}_{t+\Delta t} | \mathbf{F}\mathbf{y}_t + \mathbf{f}, \Sigma_u)$$

which yields a Gaussian distribution with mean and covariance

$$\hat{\boldsymbol{\mu}}_t = \begin{bmatrix} \boldsymbol{\mu}_t \\ \mathbf{F}\boldsymbol{\mu}_t + \mathbf{f} \end{bmatrix}, \quad \hat{\Sigma}_t = \begin{bmatrix} \Sigma_t & \Sigma_t \mathbf{F}^T \\ \mathbf{F}\Sigma_t & \mathbf{F}\Sigma_t \mathbf{F}^T + \Sigma_s \Delta t. \end{bmatrix} \quad (2.42)$$

The noise covariance Σ_s is obtained by matching both covariance matrices given in Equation (2.41) and (2.42),

$$\begin{aligned} \Sigma_s \Delta t &= \Sigma_{t+\Delta t} - \mathbf{F}\Sigma_t \mathbf{F}^T = \Sigma_{t+\Delta t} - \mathbf{F}\Sigma_t \Sigma_t^{-1} \Sigma_t \mathbf{F}^T \\ &= \Sigma_{t+\Delta t} - \mathbf{C}_t^T \Sigma_t^{-1} \mathbf{C}_t, \end{aligned} \quad (2.43)$$

and solving for Σ_s . The variance Σ_u of the control noise is then given by

$$\Sigma_u = B^\dagger \Sigma_s B^{\dagger T}. \quad (2.44)$$

The variance of our stochastic feedback controller does not depend on the controller gains and can be pre-computed before estimating the controller gains. If the computed desired control noise is smaller than the real control noise of the system, we use the control noise of the system to calculate the feedback gain matrix \mathbf{K} . Otherwise the estimated Σ_u is used to allow the trajectory distribution to increase its variance.

Controlling a Physical System

On a non-linear physical system, we first obtain the linearization of the dynamics model using the current state \mathbf{y}_t and use this linearization to obtain the parameters of the controller for the current time step in an online manner.

For a physical system, we also have to consider that the variance of the control noise Σ_u , computed from Equation (2.44), contains two sources of noise; first, the inherent system noise Σ'_u , and, second, the additional noise injected into the system by the demonstrator. Therefore, if we apply the control noise Σ_u the inherent system noise will still be present and, as a result, our controller will not match the demonstrated

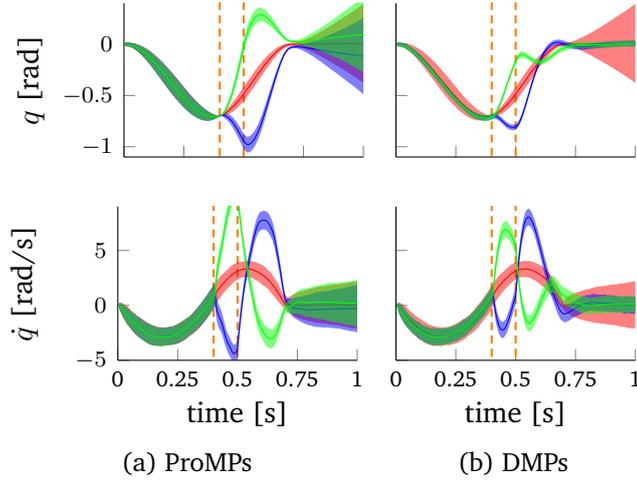


Figure 2.6: Robustness evaluation. We applied a perturbation between the dashed lines with an amplitude of $P = 200(m/s^2)$ (green), or an amplitude of $P = -200(m/s^2)$ (blue). The ProMPs (a) show compliant behavior but pass through the via-point accurately. The DMPs (b) are much stiffer and compensate the perturbation faster, before the via-point was reached. The DMPs exhibit a less efficient recovery strategy due to the higher actions.

distribution as it already contained the system noise. Therefore, we compute the control noise covariance

$$\Sigma_u^{[\text{new}]} = \Sigma_u - \Sigma'_u \quad (2.45)$$

by subtracting the estimated system noise Σ'_u from the controller noise Σ_u , computed from Equation (2.44). If the resulting controller noise is not positive definite, e.g., when the system noise estimate is higher than the control noise, we set the control noise to zero.

Illustrative Example - Robustness Analysis. In order to evaluate the robustness of our approach, we test different MP approaches under strong perturbation occurring during the execution of the movement, see Figure 2.6. Our control approach demonstrates compliant behavior when the variance of the movement is high. It allows larger deviations from the demonstrated distribution and takes more time to “return” to the distribution. However, it manages to pass accurately through the via-points as this point has small variance. The DMPs on the other hand, use high feedback gains which results in a less compliant movement which quickly tries to return to the mean trajectory. Such strategy results in unnecessary high control actions as DMPs do not have a notion of the importance of time points.

Relation to Optimal Control

Our controller derivation has strong relations to optimal control (OC). Equation (2.34) resembles a continuous time Riccati equation that is typically used for state estimation (Todorov, 2008), only the observation noise is missing as it is not present in our application. It is well known that state estimation and optimal control

are dual problems that can be solved in the same framework (Todorov, 2008). Yet, our usage of the Ricatti equation is quite different from OC and state estimation. Both approaches use the Ricatti equation for backwards integration of the value function, or the covariance, respectively. In contrast, we assume that the covariance and its derivative are already known. In this case, we can use the Ricatti equation to obtain the controller gains and no backwards integration is required. By circumventing the backwards integration, we can also avoid limitations of many OC algorithms. Almost all OC methods require a linearization of the model along a nominal mean trajectory. Using this linearization, an approximately optimal *linear* controller can be obtained (Li and Todorov, 2004; Toussaint, 2009). In contrast, our ProMP controller is non-linear as the linearization of the system is computed online for the current state. The use of OC or state estimation would also require that we know either the reward function or the observation model. Both quantities are unknown in the imitation learning scenario.

2.5 Experiments

We evaluate our approach on simulated and real robot experiments. Our experimental setups cover several aspects of our framework, i.e., stroke-based and rhythmic movements, linear and non-linear systems, simple trajectory following tasks, coordinated movements, and complex experiments such as table tennis or robot hockey.

For the real-robot experiments, i.e., the Astrojax, the maracas and the hockey task, we gathered demonstrations by kinesthetic teach-in, whereas for the simulated tasks we specify a cost function for finding the optimal time-varying controller. We used the optimal control algorithm from (Toussaint, 2009). For stroke-based movements, we train our approach as in Sec. 2.4.2 and for periodic tasks we use the EM approach in Sec. 2.4.2. An overview of the experiments performed and their objectives is given in Table 2.3. The open parameters of our approach were hand-picked and no further tuning was necessary.

2.5.1 7-link Reaching Task

In this task, we use a seven link planar robot that has to reach desired target positions in task-space, at different time points, with its end-effector. Our goal is to demonstrate the co-activation of ProMPs to solve a combination of tasks by combining two different movements. In addition, the task evaluates the necessity of the coupling between the joints of the robot, which is implemented by the ProMPs. As many joint configurations can lead to the same end-effector position, the end-effector of the robot can exhibit high accuracy, whereas each individual joint can exhibit higher variability. In this experiment, the end-effector has low variability at the task-space via-points. In order to successfully reproduce the demonstrated movements, ProMPs must correctly capture and reproduce the coupling between the DoF of the robot.

In the first set of demonstrations, the robot has to reach the via-point at $t_1 = 0.25s$. The reproduced behavior with the ProMPs is illustrated in Figure 2.7a(top). We learned the coupling of all seven joints with one ProMP. The ProMP exactly reproduced the via-points in task space while exhibiting a large variability for time steps

Table 2.3: Overview of the experimental evaluation of ProMPs

Experiment	Real robot	#DoF	Basis Type	#Demos	#Basis	Evaluation Objectives
7-link Reach.	Sim.	7	Gaussian	200	20	movement coordination, adaptation to via-points, combination
Double Pend.	Sim.	2	Gaussian	100	36	non-linear system, change in the dynamics
Astrojax	✓	7	Von-Mises	7 periods	30	periodic movements, movement coordination
Maracas	✓	7	Von-Mises	5 periods	10	periodic movements, temporal modulation, blending
Hockey	✓	7	Gaussian	10+10	10	union, combination, conditioning, context
Table Tennis	Sim.	7	Gaussian	20	15	generalization in a complex noisy environment

in between the via-points. Moreover, the ProMP could also reproduce the coupling of the joints from the optimal control law which can be seen by the small variance of the end-effector in comparison to the rather large variance of the single joints at the via-points. The ProMP achieved an average cost value of similar quality as the optimal controller.

In the second set of demonstrations the first via-point was located at time step $t_2 = 0.75s$. The movement of the robot is illustrated for specific time steps in Figure 2.7a(middle). We combined both primitives and the resulting movement is illustrated in Figure 2.7a(bottom). The combination of both MPs accurately reaches both via-points at $t_1 = 0.25$ and $t_2 = 0.75$, generating a primitive that satisfies *both* tasks.

Moreover, we evaluated the reproduction cost our approach to the number of training demonstrations in Figure 2.7b. The comparison was performed on the first set of demonstrations, i.e. top row of Figure 2.7a. With only two training demonstrations, our approach depends heavily on the regularization coefficients for the estimation of the covariance matrix and, on average, produces higher actions compared to using more demonstrations for training. In Figure 2.7b, we show that the performance of our approach does not significantly improve using more than 20 demonstrations for training. Additionally, we evaluated the performance of the inverse covariance controller (Calinon et al., 2010b) and the DMPs (Ijspeert et al., 2003). The cost for every experiment is averaged over 200 reproductions. Additionally, we average over 10 trials, where for each trial, we randomly regenerated the demonstrations using an optimal control law.

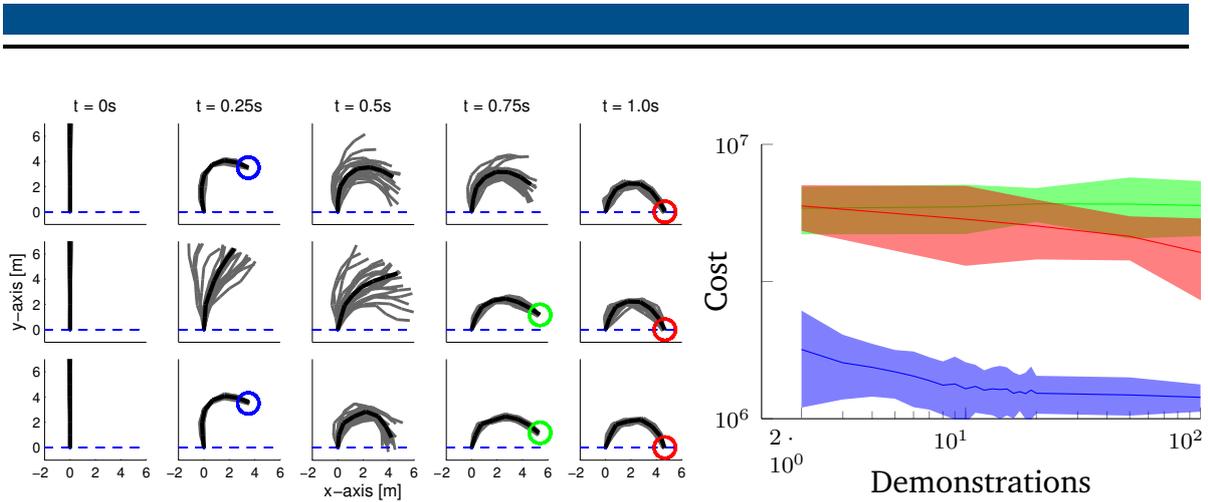


Figure 2.7: (a) A 7-link planar robot has to reach a target position at $T = 1.0s$ with its end-effector while passing a via-point at $t_1 = 0.25s$ (top) or $t_2 = 0.75s$ (middle). The plot shows the mean posture of the robot at different time steps in black and samples generated by the ProMP in gray. The ProMP approach was able to exactly reproduce the demonstration which have been generated by an optimal control law. The combination of both learned ProMPs is shown in the bottom. The resulting movement reached both via-points with high accuracy. (b) Evaluation of the reproduction cost versus the number of demonstrations provided for training on the 7-link task-space via-point task. We present the results using ProMPs (blue), the Inv. Cov. Ctl. (red) (Calinon et al., 2010b), and DMPs (green) (Ijspeert et al., 2003). The cost is averaged over 200 reproductions for every approach and over 10 trials.

2.5.2 Double Pendulum

In this experiment we evaluate our control approach on a system with non-linear dynamics. We use a simulated double-pendulum with unit link lengths and unit masses. Non-linearities are induced due to gravity, centripetal and Coriolis forces. During the execution of our controller we compute a linearization of the system dynamics at every time step at the state \mathbf{y}_t to obtain $\{\mathbf{A}_t, \mathbf{B}_t, \mathbf{c}_t\}$.

In this experiment, we also evaluate the robustness of the controller to changes in the system dynamics. To this end, we generated demonstrations on a linear double-link system, i.e. without gravity, centripetal, and Coriolis forces taken into account, using the optimal controller. Subsequently, we executed the learned trajectory distribution on the non-linear dynamical system using the ProMP controller that uses the linearization of the real dynamics. The linearization is performed in an online manner at the current state of the system for each of the reproductions, resulting in state-dependent gains and a non-linear control architecture. Our results are presented in Figure 2.8. The reproduced trajectory distribution matches the demonstrations, despite the drastic change in the dynamics of the system. Additionally, we compare to the ProMP controller if we use a pre-linearization of the system dynamics along the mean trajectory, which is given in Figure 2.8 (second row). Linearizing at the mean

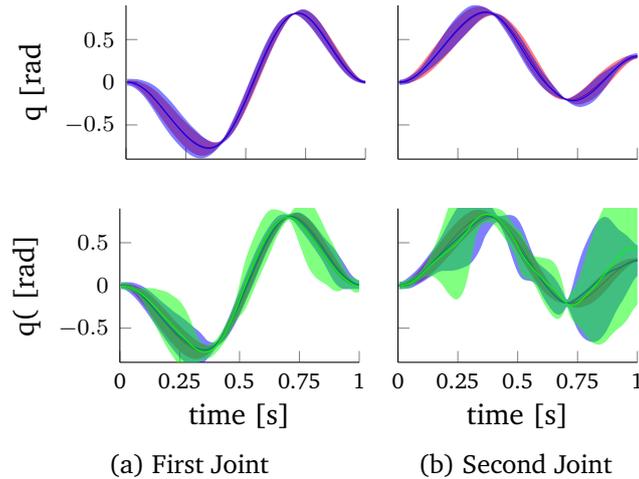


Figure 2.8: *Double pendulum, non-linear system.* In red we depict the demonstrated trajectory distribution. (first row) In this experiment, we use the optimal controller to generate demonstrations on a linear system. Subsequently, we executed our controller on a non-linear double-pendulum system. The reproduced trajectory distribution (blue) match the demonstrations (red) despite the changed dynamics. The ProMP controller is using the linearization at the current state to compute the control gains. (second row) We illustrate the performance of our approach by using *non* state-independent gains (blue) where the linearization is performed offline along the mean state trajectory. As can be seen, ProMPs with state-independent gains are not capable of reproducing the demonstrated trajectory distribution. In green, we evaluate the performance of a linearized version of the non-linear ProMP controller which has been learned by fitting a linear model to the data produced by the ProMP controller. Also the linearized ProMP controller fails at tracking the distribution, showing that the state-dependent gains of the ProMP controller that cause the non-linearity are essential for accurate tracking in non-linear systems.

trajectory results in a linear feedback controller with state-independent gains and, hence, the resulting controller can not reproduce the demonstrated trajectory distribution. Moreover, we evaluated the reproduction a learned linear Gaussian controller per time-step which is learned from data obtained from the ProMP controller. We used the ProMP reproductions as our classical optimal control method (Toussaint, 2009) failed to find a solution that was minimizing the given cost function. This approach is a linearized version of the non-linear ProMP controller. Our results in Figure 2.8 show that the tracking performance reduces significantly, which proofs that the non-linearities of the ProMP controller are essential for accurate distribution tracking in non-linear systems.

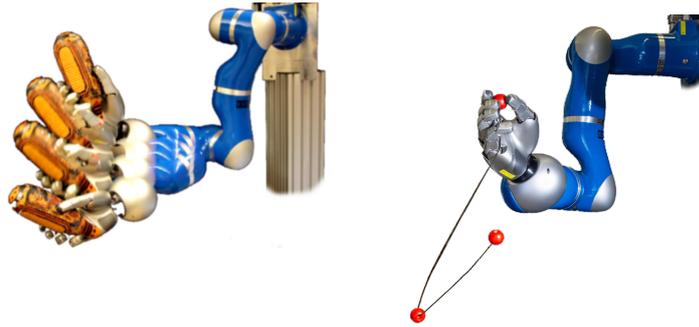


Figure 2.9: Two real robot setups that we used for the evaluation of our approach. (left) The KUKA arm playing the maracas musical instrument. We demonstrated a slow version of the rhythmic shaking movement and we progressively increased the speed. (right) The KUKA arm playing with an Astrojax. The robot learned the game from demonstrations.

2.5.3 Playing Astrojax

‘Astrojax’ is a toy consisting of three balls on a string. Two balls are fixed at either end of the string, while one ball is free to slide along the string. Roughly, ‘Astrojax’ is a game between ‘YoYo’ and juggling. In order to successfully play ‘Astrojax’, the bottom two balls should orbit each other and not get in touch. We use the ‘Astrojax’ experiment to demonstrate that ProMPs can successfully learn and reproduce periodic movements. The real-robot setup is shown in in Figure 2.9 and Figure 2.10. The hand performs a stable grasp and is not controlled by ProMPs. We demonstrate a rhythmic movement to the robot which created a “basic orbit” pattern. We subsequently use the ProMPs to learn the movement with thirty Von-Mises basis functions for each joint. The robot could reproduce the behavior and recreated the same pattern, as illustrated in Figure 2.10. The demonstrations exhibit a lot of variability and the robot generate periodic movements which show the same type of variability. During the demonstrations, we were capable of sustaining a successful orbit of the ‘Astrojax’ for a mean duration of $t_{\text{demo}} = 8.2(s)$. During the reproduction, we achieved a mean orbiting of $t_{\text{reprod.}} = 15.2(s)$. In contrast, the DMP approach would repeat always exactly the same movement, rendering the behavior different than the demonstrated one. DMPs are neither capable of reproducing variability, be compliant, or generate coordinated movements. GMR approaches, to our knowledge, have not yet investigated the application in periodic movements. A video with the robot playing ‘Astrojax’ can be found at <http://www.ausy.tu-darmstadt.de/uploads/Team/AlexandrosParaschos/Astrojax.mp4>.

2.5.4 Robot Maracas

The maracas is a musical instrument containing grains. Shaking the maracas produces sounds. We used the KUKA lightweight arm for the experiments and the DLR hand to grasp the instrument. The hand was only used for holding the maracas and was not controlled by the ProMPs. Our setup is shown in Figure 2.9.

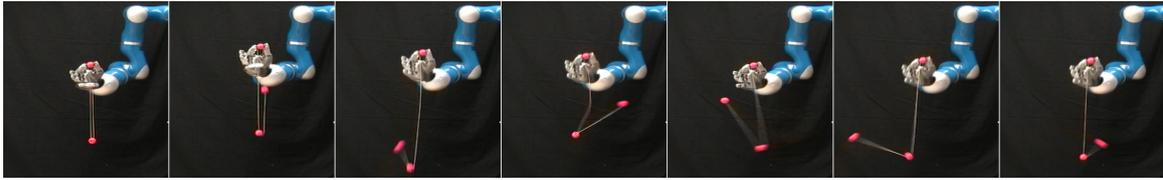


Figure 2.10: The KUKA light-weight arm playing “Astrojax”. The robot holds one of the balls in his fingers and starts with releasing the ball that is connected to the other end of the string. It subsequently reproduces the demonstrated rhythmic movement showing the same human-like variability in its movement pattern.

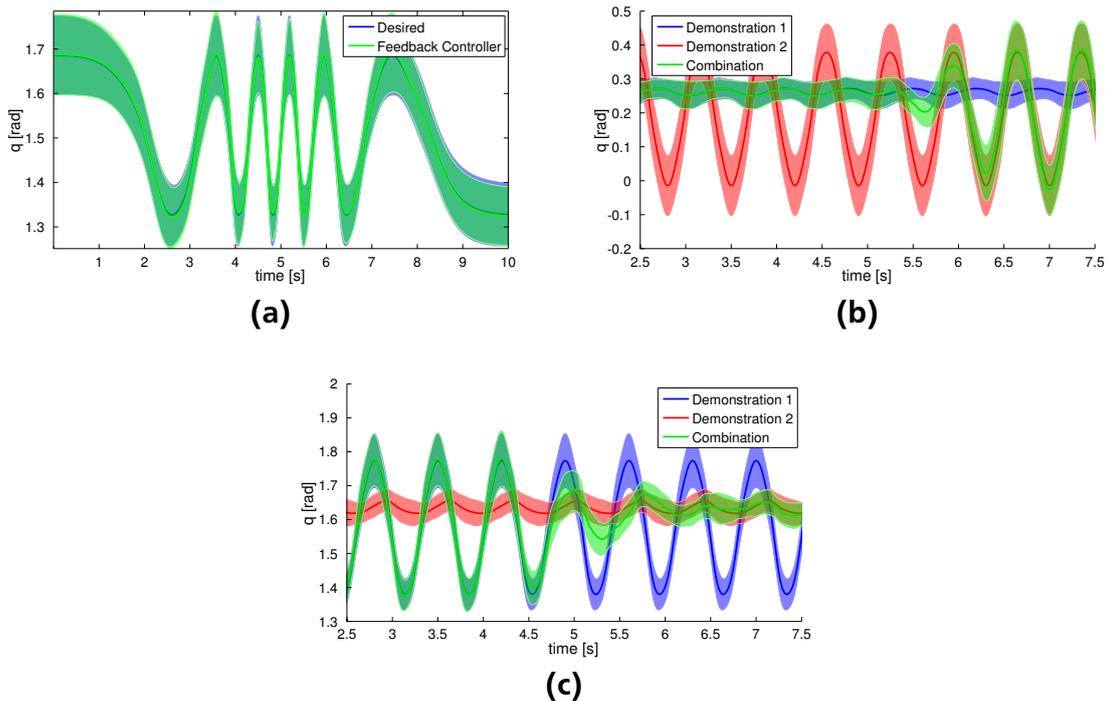


Figure 2.11: (a) The trajectory distribution of the fourth joint when playing maracas. The speed of the movement is adapted by modulating the speed of the phase signal z_t . The desired distribution is shown in blue and the generated distribution from the feedback controller in green. Both distributions match. (b,c) Blending between two rhythmic movements (blue and red areas). The green area is produced by continuously switching from the blue to the red movement.

As demonstrating fast movements with kinesthetic teach-in can be difficult on the real robot arm due to the inertia, friction, and model discrepancies, we demonstrate a slower movement of ten periods. We used this slow demonstration for learning the primitive but modulated the speed of the phase during reproduction. The faster movement achieved a shaking movement of appropriate speed that generates the desired sound of the instrument.

We learned the rhythmic movement using $N = 10$ Von-Misses basis functions per dimension. The ProMP was trained all seven DoF of the robot. We optimized the parameters of ProMPs using the Expectation Maximization algorithm. To do so, we split the demonstration in $M = 400$ segments and assigned the appropriate phase signal. We executed our controller after training and we measured that the generated trajectories stay on average 94.4% of the total time within two standard deviations of demonstrated distribution. After learning the ProMP model from the demonstration, we progressively increase the speed of the movement by modulating the phase, such that the robot successfully plays the instrument.

The speed of the motion can be changed during execution to achieve different sound patterns. We show an example movement of the robot in Figure 2.11(a). The desired trajectory distribution of the demonstrated rhythmic movement and the resulting distribution generated from the feedback controller again match.

Additionally, we demonstrated a second type of rhythmic shaking movement and use it to continuously blend between both movements to produce different sounds. One such transition between the two ProMPs is shown for one joint in Figure 2.11(b) and (c). We measured the trajectory reproduction accuracy from our controller against the desired blended distributions and found that the trajectories are within two standard deviations for 92.7%, and 93.4% of the total execution time, respectively. A video showing the demonstration phase, reproduction with time modulation, and blending two primitives can be found at <http://www.ausy.tu-darmstadt.de/uploads/Team/AlexandrosParaschos/Maracas.mp4>

2.5.5 Robot Hockey

In the hockey task, the robot has to shoot a hockey puck in different directions and for different distances. The task setup is depicted in Figure 2.12(a). We used the KUKA lightweight arm for this experiment and controlled the accelerations of the arm with the ProMPs using an inverse dynamics controller. The control parameters of the robot $t_{k \in 1 \dots K}$ are the desired position vector $\mathbf{q}_t \in \mathbb{R}^7$ and the desired acceleration $\ddot{\mathbf{q}}_t \in \mathbb{R}^7$ of each joint. The ProMPs provide at every time point the desired acceleration $\ddot{\mathbf{q}}_t$, while the desired position \mathbf{q}_t is obtained from second-order Euler integration of the acceleration. The duration of the control step is $\Delta t = 1\text{ms}$. A hockey stick is mounted as an end-effector for hitting the puck.

We again used two sets of demonstrations. The first set contained $M_1 = 10$ demonstrations where the robot shot the puck straight at varying distances. The demonstrations were provided by a human tutor, using kinesthetic teaching. The second set also contained $M_2 = 10$ demonstrations where the demonstrator shot the puck at varying angles, while trying to keep the variance of the distance relatively small. For both demonstration sets, we trained two ProMPs using $N = 10$ Gaussian basis func-

tions per dimension, which resulted in a weight vector $w \in \mathbb{R}^{70}$. By reproducing the learned primitives, we obtain behaviors illustrated in Figure 2.12(b) and Figure (c) respectively. The shots exhibit the demonstrated variability in either angle or distance. We generated the images in Figure 2.12 by taking the picture of the robot’s configuration after the execution of the primitive and the puck has stopped. The figures show an overlay of the images from multiple executions of each primitive. By training a primitive on the union of the two datasets, the robot is able to shoot the puck at a variety of angles and distances, as illustrated in Figure 2.12(d). Additionally, we co-activated the two individual primitives and the resulting MP shoots only in the center at medium distance, i.e., the intersection of both MPs, as illustrated in Figure 2.12(e). This experiment again illustrates the achievement of a combination of tasks, where the first task was to shoot at a desired angle and the second, to shoot at a desired distance.

Finally, we learned a conditional distribution over the trajectories conditioned on the angle of the final puck position as described in Sec. 2.4.3. The resulting primitive was able to shoot at the desired angle as illustrated in Figure 2.12(f). All the operations are computed in closed form, no re-estimation of the primitive parameters is needed to compute the generalization or the combination of the primitives.

We provide a cost function evaluation of the two demonstrated datasets, the “angle” and the “distance” dataset, and the respective reproduction in Table 2.4. The cost function is chosen intuitively to resemble the desired task. By giving the human demonstrator a specific task, we can assume that he is minimizing a similar cost function, at least in approximation. Our approach successfully reproduces the same costs as in the demonstrations. The cost function of the “distance” dataset contains demonstrations that shoot the puck at different distances, but aiming at the same angle. Therefore, it only penalizes deviations from the desired angle. Similarly, in the “angle” dataset, the cost function penalizes deviations from the desired distance. Since, shooting the puck at a specific distance is quite hard due to different environment variables, i.e, friction between the puck surface and the floor, we choose a lower deviation penalty.

We also evaluated the cost on the combined movement which is supposed to solve both tasks, i.e., shoot at a specific distance and angle. For this evaluation, we added the cost functions from the “distance” and “angle” datasets. In Table 2.4, we show that the reproduction of the combination, which is a newly composed behavior not present in the demonstrations, achieves significantly lower costs than both original datasets.

2.5.6 Simulated Table Tennis

In this experiment, we evaluate the generalization capabilities of the ProMPs for a complex task. As comparison, we use the DMP approach presented in (Kober et al., 2010). The robot, a simulated BioRob 5-DoF arm (Klug et al., 2008), is mounted on two linear axis and equipped with an additional shoulder joint. The setup is shown in Figure 2.13a. We control the robot with inverse dynamics control. We used an imperfect inverse dynamics model to render the simulation more realistic. As a result, the desired and actual trajectories do not match exactly and, thus, make the

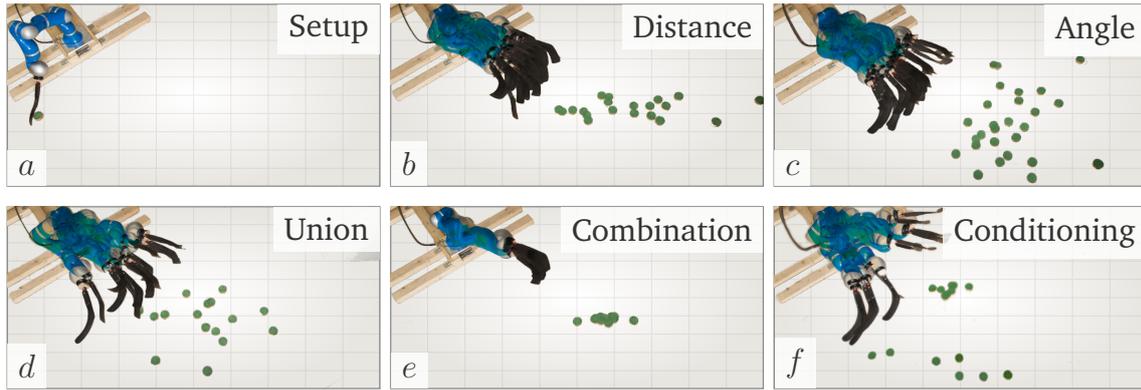


Figure 2.12: Robot Hockey. The robot shoots a hockey puck. The figure shows overlaid images of the real-robot setup that is set on the flour, taken from above. We demonstrate ten straight shots with varying distances and ten shots with varying angles. The pictures show samples from the ProMP model for straight shots (*b*) and shots with different angles (*c*). Learning from the union of the two data sets yields a model that represents variance in both distance and angle (*d*). Co-activating the individual MPs leads to a combined MP that reproduces shots where both models had probability mass, i.e., in the center at medium distance (*e*). The last picture shows the effect of conditioning on the angle of the shoot (*f*).

robot more sensitive to jerky movements as jerky movements are harder to track. At the beginning of each experiment, the ball is set to different pre-specified positions and initial velocities.

The robot has to return the ball to a specific target area at the opponents field. For this experiment, we gathered trajectories for 15 different combinations of initial ball configurations and robot targets, generated from an analytical player (Muelling et al., 2011). We trained the ProMP approach with the whole data-set and created a single primitive. In our experiment, the ball state is set at the beginning of a trial and the ProMP is conditioned to the predicted hitting position and velocity in joint space, obtained from the analytical player. A delay before the start of the execution of the primitive is provided by the simulation. In order to make the task more realistic, we assume that the ball state is estimated, instead of being directly observed, with zero-mean i.i.d. Gaussian noise. The noise on the ball position increases the task difficulty significantly as it also affects the estimated time until the ball reaches the hitting plane. We evaluate the ProMPs and the DMPs on each of the 15 task setups by computing the average distance to the target and the average success rate. We display our results on Figure 2.13b.

The DMP was trained with only one demonstration, while the goal position and velocity were modified according to predicted hitting point using the approach presented in (Kober et al., 2010). The DMP had inferior performance as it significantly deforms the trajectories, which makes the resulting trajectory harder to track as the feedback controller saturates in torque limits due the deformation. This saturation has the effect that the robot does not reach the specified hitting point with the specified velocity.

Table 2.4: Evaluation of the average cost for the Robot Hockey experiment. We present the average cost of the human demonstrations for both demonstrated datasets. The robot reproduction results in similar cost as the demonstrations. The “Combination” cost is specified as the sum of both cost functions. The robot produces a novel composed behavior that performs significantly better than both demonstrated sets.

Dataset		Average Cost
Demonstrations	Distance	1.20 ± 1.18
	Angle	2.21 ± 2.95
Reproduction	Distance	1.24 ± 1.24
	Angle	2.07 ± 3.16
	Combination	2.52 ± 1.59
Evaluation	Dist. on Comb.	6.21 ± 8.18
	Angle on Comb.	25.97 ± 21.54

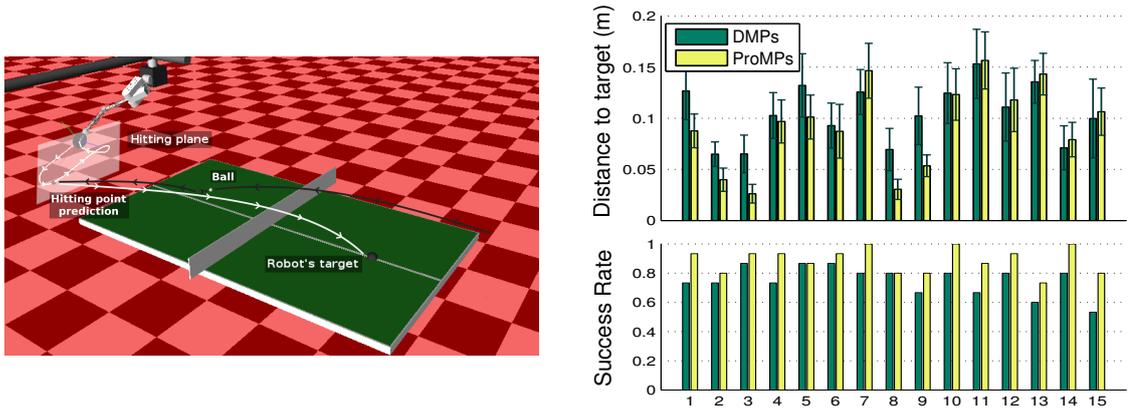


Figure 2.13: (a) The simulated table tennis setup. We indicate the robot arm mounted on linear axis, the ball position, the hitting plane in which the robot will try to hit the ball, and the hitting point prediction. Due to the induced noise in our simulation the desired and actual hitting points may differ. On the opponent’s side, we can see the robot’s target for this simulation. In our experiments, we use 15 different combinations of initial ball positions and targets covering most of the table. (b) The distance between the impact position of the ball on the opponents field and the actual targeted point in meters, for the DMP and the ProMP approaches. We average the results over 20 samples where Gaussian observation noise was added to the initial ball position. The bars denote the mean error and the error-bars one standard deviation. (bottom) Shows the success rate for each combination. If the distance between the landed position and the target position is less than 0.4 meters it is counted as a success. The performance of ProMPs is superior in all the experiments leading generally to smaller errors with an increased success rate.

2.6 Epilogue

Probabilistic movement primitives are a promising approach for learning, modulating, and re-using movements in a modular control architecture. To effectively take advantage of such a control architecture, ProMPs support simultaneous activation, match the quality of the encoded behavior from the demonstrations, are able to adapt to different desired target positions, and can be efficiently learned by imitation. In ProMPs we parametrize the desired trajectory distribution of the primitive by a hierarchical Bayesian model with Gaussian distributions. The trajectory distribution can be easily obtained from demonstrations and simultaneously defines a feedback controller which is used for movement execution. Our probabilistic formulation introduces new operations for movement primitives, such as conditioning and combination of primitives. All these mechanisms do not exist for alternative representations and, with ProMPs, we provide a single mathematical framework to describe them. Future work will focus on using the ProMPs in a modular control architecture and improving upon imitation learning by reinforcement learning.

The advanced flexibility of ProMPs comes to a cost of requiring multiple demonstrations in order to accurately encode the distribution over the trajectories. The number of demonstrations required depend on the complexity of the task and, from our experience, $\sim 10 - 20$ suffice for simple tasks. Prior knowledge about the task can be incorporate by using prior distributions and regularization techniques. Furthermore, our approach is appropriate for tasks that have a strong coupling to time. For tasks loosely coupled with time, other approached might produce better results. Finally, it should be noted that our approach can not capture multiple modes since we only use a single Gaussian component to encode the trajectory distribution.

3 Model-Free Probabilistic Movement Primitives

3.1 Introduction

Developing robots that operate in the same environment with humans and physically interacting with every-day objects requires accurate control of the contact forces that occur during the interaction. While non-compliant robots can achieve a great accuracy, the uncertainty of complex and less-structured environment prohibits physical interaction. In this chapter, we focus on providing a compliant control scheme that can enable robots to manipulate their environment without damaging it. Typically, force-control requires an accurate dynamics model of the robot and its environment that is not easy to obtain. Other approaches suggest to learn a dynamics model, however, this process can be time-consuming and is prone to model-errors. We present an approach that can jointly learn the desired movement of the robot and the contact forces by human demonstrations, without relying on a learned forward or inverse model.

Existing approaches for motor skill learning that are based on movement primitives (Ijspeert et al., 2003; Schaal et al., 2005; Billard et al., 2008; Kober and Peters, 2009; Ijspeert et al., 2013), often incorporate into the movement primitive representation the forces needed for the physical interactions (Pastor et al., 2011; Kalakrishnan et al., 2011; Gams et al., 2014). However, such approaches model a single successful reproduction of the task. Multiple demonstrations are typically averaged, despite that they actually represent similar, but different, solutions of the task. Thus, the applied contact forces are not correlated with the state of the robot nor sensory values that indicate the state of the environment, e.g., how heavy an object is.

We propose learning the coordination of the interaction forces, with the kinematic state of the system, as well as the control actions needed to reproduce the movement exclusively from demonstration. Motor skill learning for such interaction tasks for high-dimensional redundant robots is challenging. This task requires real-time feedback control laws that process sensory data including joint encoders, tactile feedback and force-torque readings. We present a model-free version of the Probabilistic Movement Primitives (ProMPs) (Paraschos et al., 2013a) that enables robots to acquire complex motor skills from demonstrations, while it can coordinate the movement with force, torque, or tactile sensing. The ProMPs have several beneficial properties, such as generalization to novel situations, combination of primitives and time-scaling, which we inherit in our approach.

ProMPs assume a locally linearizable dynamics models to compute time-varying feedback control laws. However, such dynamics models are hard to obtain for physical interaction tasks. Therefore we obtain a time varying feedback controller directly

from the demonstration without requiring such a model. In the model-free extension of the ProMPs, we condition the joint distribution over states and controls on the current state of the system, and obtain a distribution over the controls. We show that this distribution represents a time-varying stochastic linear feedback controller. Due to the time-varying feedback gains, the controller can exhibit behavior with variable stiffness and, thus, it is safe to use in physical interaction. A similar control approach has recently been presented in (Lee et al., 2015).

Our approach inherits many beneficial properties of the original ProMP formulation. We can reproduce the variability in the demonstrations and use probabilistic operators for generalization to new tasks or the co-activation of learned primitives. The resulting feedback controller shows similar properties as in the model-based ProMP approach, it can reproduce optimal behavior for stochastic systems and exactly follow the learned trajectory distribution, at least, if the real system dynamics are approximately linear for each time step. For non-linear systems, the estimated variable stiffness controller can get unstable if the robot reaches configurations that are far away from the set of demonstrations. To avoid this problem, we smoothly switch between a stable PD-controller and the ProMP controller if the support of the learned distribution for the current situation is small. We show that this extension allows us to track trajectory distributions accurately even for non-linear systems.

In this chapter, we extend our approach (Paraschos et al., 2015) to provide a more detailed explanation on sensory integration, introduce a mixture model of primitives, present how our approach can be used for adapting the interaction forces to user’s input, and evaluate our approach in more complex robotic tasks.

3.2 Related Work

In this section, we review related work on movement primitives for imitation learning that combine position and force tracking, model the coupling between kinematics and forces and are able to capture the correlations between these two quantities.

The benefit of an additional feedback controller to track desired reference forces was demonstrated in grasping tasks in (Pastor et al., 2011). Individual dynamical systems (DMPs) (Ijspeert et al., 2013) were trained for both, position and force profiles in imitation learning. The force feedback controller substantially improved the success rate of grasps in tracking demonstrated contact forces under changing conditions. For manipulation tasks like opening a door, the authors showed that the learned force profiles can be further improved through reinforcement learning (Kalakrishnan et al., 2011). However, the approach requires the manual tuning of the gains and has limited generalization capabilities of the movement (Paraschos et al., 2013b). This approach is applicable for tasks where learning a single reference force profile suffices and generalization to new situations assume that the system dynamics stay constant.

For many tasks, such as like bi-manual manipulations, the feedback controller needs to be coupled. Gams et al. (Gams et al., 2010) proposed *cooperative* dynamical systems, where deviations from desired forces modulate the velocity forcing term in the DMPs for position control. This approach was tested on two independently operating robot arms solving cooperative tasks like lifting a stick (Gams et al., 2014). Deviations in the sensed contact forces in one robot were used to adapt the DMP of

the other robot and the coupling parameters were obtained through iterative learning control. A related probabilistic imitation learning approach to capture the couplings in time was proposed in (Kormushev et al., 2011b). In this approach, Gaussian mixture models were used to represent the variance of the demonstrations. For training this approach the robot first reproduces the learned positional movement and then, with the help of an external force input device, the force-profile is learned. The position and force profiles are coupled only in time and cross-correlations are not captured. The approach was evaluated successfully on complex physical interaction tasks such as ironing, opening a door, or pushing against a wall.

Adapting Gaussian Mixture Models (GMMs) (Calinon et al., 2010a; Kormushev et al., 2011a; Kronander and Billard, 2013; Calinon et al., 2014) have been proposed for use in physical interaction tasks. The major difference to the dynamical systems approach is that GMMs can represent the variance of the movement. Closely related to our approach, Evrard et al. in (Evrard et al., 2009) used GMMs to learn joint distributions of positions and forces. Joint distributions capture the correlations between positions and forces and were used to improve adaptation to perturbations in cooperative human robot tasks for object lifting. In this approach, the control gains were fixed to track the mean of the demonstrated trajectories. In (Gribovskaya et al., 2011), it was shown that by assuming known forward dynamics, variable stiffness control gains can be derived in closed form to match the demonstrations. We address here an important related question of how these gains can be learned in a model-free approach from the demonstrations.

3.3 Model-Free Probabilistic Movement Primitives

We propose a novel framework for robot control which can be employed in physical interaction scenarios. In our approach, we jointly learn the desired trajectory distribution of the robot’s joints or end-effectors and the corresponding controls signals. We train our approach from a limited set of demonstrations. We refer to the joint distribution as state-action distribution. Further, we incorporate proprioceptive sensing, such as force or tactile sensing, into our state representation. The additional sensing capabilities are of high importance for physical interaction as they can disambiguate kinetically similar states. We present our approach by, first, extending the Probabilistic Movement Primitives (ProMPs) framework (Paraschos et al., 2013a) to encode the state-action distribution and, second, we derive a stochastic feedback controller without the use of a given system dynamics model. Finally, we extend our control approach for states which are relatively far from the vicinity of the learned state-action distribution. In that case, our control approach can no longer produce correcting actions and an additional backup controller with high gains is needed. Our framework inherits most of the beneficial properties introduced by the ProMPs that significantly improved generalization to novel situations and enables the generation of primitives that *concurrently* solve multiple tasks (Paraschos et al., 2013a).

3.3.1 Encoding the Time-Varying State-Action Distribution of the Movement

We avoid explicitly learning robot and environment models by learning directly the appropriate control inputs, while keeping the beneficial properties of the ProMP approach, such as generalization and concurrent execution.

In order to simplify the illustration of our approach, we first discuss the special case of a single Degree of Freedom (DoF) and, subsequently, we expand our description to the generic case of multiple DoF. The description is based in (Paraschos et al., 2013a), but modified appropriately to clarify how the actions can be modelled. First, we define the extended state of the system as

$$\mathbf{y}_t = [q_t, \dot{q}_t, u_t]^T, \quad (3.1)$$

where q_t is the position of the joint, \dot{q}_t the velocity, and u_t the control applied at time-step t . Similar to ProMPs, we use a linear basis function model to encode the trajectory of the extended state \mathbf{y}_t . The feature matrix and the weight vector of the non-linear function approximation model become

$$\mathbf{y}_t = \begin{bmatrix} q_t \\ \dot{q}_t \\ u_t \end{bmatrix} = \tilde{\Phi}_t \mathbf{w}, \quad \tilde{\Phi}_t = \begin{bmatrix} \phi_t^T & \mathbf{0} \\ \dot{\phi}_t^T & \mathbf{0} \\ \mathbf{0} & \psi_t^T \end{bmatrix}, \quad \mathbf{w} = \begin{bmatrix} \mathbf{w}_q \\ \mathbf{w}_u \end{bmatrix}, \quad (3.2)$$

where the vectors ϕ_t and ψ_t represent the feature vectors for the position q_t and the control u_t respectively. The derivative of the position feature vector $\dot{\phi}_t$ is used to compute the velocity of the joint \dot{q}_t . The weight vector \mathbf{w} contains the weight vector for the position \mathbf{w}_q and the weight vector for the control \mathbf{w}_u . The dimensionality of the feature ϕ_t and weight \mathbf{w}_q vectors is $N \times 1$, where N is the number of features used to encode the joint position. Similarly, the dimensionality of ψ_t and \mathbf{w}_u vectors is $M \times 1$. The remaining entries of $\tilde{\Phi}_t$, denoted by $\mathbf{0}$, are zero-matrices with the appropriate dimensionality. In our approach, we distinguish between the features used to encode the position from the features used to encode the control signal due to the different properties of the two signals. The distinction allows us to use of different type of basis functions, different parameters, or a different number of basis functions.

We extend our description to the multidimensional case. First, we extend the state of the system from Equation (3.2) to

$$\mathbf{y}_t = [\mathbf{q}_t^T, \dot{\mathbf{q}}_t^T, \mathbf{u}_t^T]^T, \quad (3.3)$$

where the vector \mathbf{q}_t is a concatenation of the positions of all joints of the robot, the vector $\dot{\mathbf{q}}_t$ of the velocities of the joints, and \mathbf{u}_t of the controls respectively. The feature matrix $\tilde{\Phi}_t$ now becomes a block matrix

$$\tilde{\Phi}_t = \left[\Phi_t^T, \dot{\Phi}_t^T, \Psi_t^T \right]^T, \quad (3.4)$$

where

$$\Phi_t = \left[\begin{array}{ccc|c} \phi_t^T & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \\ \mathbf{0} & \cdots & \phi_t^T & \end{array} \right], \Psi_t = \left[\begin{array}{c|ccc} \mathbf{0} & \psi_t^T & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \psi_t^T \end{array} \right], \quad (3.5)$$

define the features for the joint positions and the joint controls. Similarly to the single DoF, the features used for the joint velocities $\dot{\Phi}_t$ are the time derivatives of the features of the joint positions Φ_t . We use the same features for every DoF. The dimensionality of the feature matrices Φ_t and Ψ_t is $K \times K \cdot (N + M)$, where K denotes the number of DoF.

The weight vector \mathbf{w} has a similar structure to Equation (3.2) and, for the multi-DoF case, is given by

$$\mathbf{w} = \left[\underbrace{{}^1\mathbf{w}_q^T, \dots, {}^K\mathbf{w}_q^T}_{\text{weights for joint positions}}, \underbrace{{}^1\mathbf{w}_u^T, \dots, {}^K\mathbf{w}_u^T}_{\text{weights for joint controls}} \right]^T, \quad (3.6)$$

where ${}^i\mathbf{w}$ denotes the weight vector for joint $i \in [1, K]$.

The probability of a single trajectory $\tau = \{\mathbf{y}_t, t \in [1 \dots T]\}$, composed from states of T subsequent time steps, given the parameters \mathbf{w} , is computed by

$$p(\tau|\mathbf{w}) = \prod_t \mathcal{N}(\mathbf{y}_t | \Phi_t \mathbf{w}, \Sigma_y), \quad (3.7)$$

where we assume *i.i.d.* Gaussian observation noise with zero mean and Σ_y covariance. Representing multiple trajectories would require a set of weights $\{\mathbf{w}\}$. Instead of explicitly maintaining such a set, we introduce a distribution over the weights $p(\mathbf{w}; \boldsymbol{\theta})$, where the parameter vector $\boldsymbol{\theta}$ defines the parameters of the distribution. Given the distribution parameters $\boldsymbol{\theta}$, the probability of the trajectory becomes

$$p(\tau; \boldsymbol{\theta}) = \int p(\tau|\mathbf{w})p(\mathbf{w}; \boldsymbol{\theta})d\mathbf{w}, \quad (3.8)$$

where we marginalize over the weights \mathbf{w} . As in the ProMP approach, we use a Gaussian distribution to represent $p(\mathbf{w}; \boldsymbol{\theta})$, where $\boldsymbol{\theta} = \{\boldsymbol{\mu}_w, \Sigma_w\}$. Using a Gaussian distribution enables the marginal to be computed analytically and facilitates learning. The distribution over the weight vector $p(\mathbf{w}; \boldsymbol{\theta})$ correlates (couples) the DoFs of the robot to the action vector at every time-step t . The probability of the current state-action vector \mathbf{y}_t given $\boldsymbol{\theta}$ is computed by

$$p(\mathbf{y}_t; \boldsymbol{\theta}) = \int \mathcal{N}(\mathbf{y}_t | \Phi_t \mathbf{w}, \Sigma_y) \mathcal{N}(\mathbf{w} | \boldsymbol{\mu}_w, \Sigma_w) d\mathbf{w} = \mathcal{N}(\mathbf{y}_t | \Phi_t \boldsymbol{\mu}_w, \Phi_t \Sigma_w \Phi_t^T + \Sigma_y),$$

in closed form. We use normalized Gaussian basis functions as features. Each basis function is defined in the time domain by

$$\phi_i(t) = \frac{b_i(t)}{\sum_{j=1}^n b_j(t)}, \quad b_i(t) = \exp\left(-\frac{(t - c_i)^2}{2h}\right), \quad (3.9)$$

where c_i denotes the center of the i th basis function and h the bandwidth. The centers of the basis functions are spread uniformly in $[-2h, T_{\text{end}} + 2h]$. The number of basis functions and the bandwidth value we used, depend on the complexity of task. Typically, complex task require higher number of basis functions in order to represent them accurately.

3.3.2 Imitation Learning for Model-Free ProMPs

We use multiple demonstrations to estimate the parameters $\theta = \{\mu_w, \Sigma_w\}$ of the distribution over the weights $p(\mathbf{w}|\theta)$. First, for each demonstration i , we use linear ridge regression to estimate the parameter vector \mathbf{w}_i associated to that specific demonstration, i.e.,

$$\mathbf{w}_i = (\Phi_t^T \Phi_t + \lambda \mathbf{I})^{-1} \Phi_t^T \mathbf{Y}_i, \quad (3.10)$$

where λ denotes the ridge factor and \mathbf{Y}_i the observations of the state and action for all the time steps of that demonstration. We set λ to zero, unless numerical issues arise. Subsequently, we estimate the parameters θ from the set of weights $\{\mathbf{w}_i, i \in [1, N]\}$ using the ML estimators for Gaussians, i.e.,

$$\mu_w = \frac{1}{L} \sum_{i=1}^L \mathbf{w}_i, \quad \Sigma_w = \frac{1}{L} \sum_{i=1}^L (\mathbf{w}_i - \mu_w)(\mathbf{w}_i - \mu_w)^T, \quad (3.11)$$

where L is the number of demonstrations.

3.3.3 Integration of Proprioceptive Feedback

Additional sensory feedback integration, e.g., force-torque feedback, is beneficial for physical interaction scenarios, as we demonstrate in Section 3.4.5, because our approach can capture the correlation of the trajectory, the controls and the sensory signal. This correlation contains useful information for the reproduction of the movement.

We extend our approach to additionally contain the sensory signal \mathbf{s}_t in the state \mathbf{y}_t , i.e., Equation (3.3) becomes

$$\mathbf{y}_t = [\mathbf{q}_t^T, \dot{\mathbf{q}}_t^T, \mathbf{s}_t, \mathbf{u}_t^T]^T. \quad (3.12)$$

The derivative of the sensory signal $\dot{\mathbf{s}}_t$ is typically not included in the state. The respective basis function matrix becomes

$$\tilde{\Phi}_t = \left[\Phi_t^T, \dot{\Phi}_t^T, \mathbf{Z}_t^T, \Psi_t^T \right]^T, \quad (3.13)$$

where \mathbf{Z}_t^T denotes the basis functions used for the external sensory signal \mathbf{s}_t . In this chapter we set the basis function \mathbf{Z}_t^T similarly to Φ_t^T , but we provide a generic derivation to allow the use of other basis functions. We train our approach as shown in Section 3.3.2, solely from demonstrations. The weight vector \mathbf{w}_s is now expanded to accommodate the weights for the additional sensory feedback dimensions, i.e.,

$$\mathbf{w} = \left[\underbrace{{}^1\mathbf{w}_q^T, \dots, {}^K\mathbf{w}_q^T}_{\text{weights for joint positions}}, \underbrace{{}^1\mathbf{w}_s^T, \dots, {}^K\mathbf{w}_s^T}_{\text{weights for force/torque sensors}}, \underbrace{{}^1\mathbf{w}_u^T, \dots, {}^K\mathbf{w}_u^T}_{\text{weights for joint controls}} \right]^T, \quad (3.14)$$

hence, by learning the distribution $p(\mathbf{w})$, we can represent the correlations between the sensory signal and the control commands. We use the sensory signal to get a new desired trajectory distribution and its controls.

3.3.4 Generalization with Conditioning

The modulation of via-points and final positions is an important property of any MP framework to adapt to new situations. Generalization to different via-points or final targets can be implemented by conditioning the distribution at reaching the desired position \mathbf{q}_t^* at time step t .

By applying Bayes theorem, we obtain a new distribution $p(\mathbf{w}|\mathbf{q}_t^*)$ for \mathbf{w} which is Gaussian with mean and variance

$$\boldsymbol{\mu}_w^{[\text{new}]} = \boldsymbol{\mu}_w + \mathbf{Q}_t (\mathbf{q}_t^* - \Psi_t^T \boldsymbol{\mu}_w), \quad (3.15)$$

$$\boldsymbol{\Sigma}_w^{[\text{new}]} = \boldsymbol{\Sigma}_w - \mathbf{Q}_t \Psi_t^T \boldsymbol{\Sigma}_w, \quad (3.16)$$

$$\mathbf{Q}_t = \boldsymbol{\Sigma}_w \Psi_t (\boldsymbol{\Sigma}_q^* + \Psi_t^T \boldsymbol{\Sigma}_w \Psi_t)^{-1}, \quad (3.17)$$

where $\boldsymbol{\Sigma}_q^*$ is a covariance matrix specifying the accuracy of the conditioning. By conditioning to the desired position, the weight vectors for the controls \mathbf{w}_u are modulated as well. Therefore, the proposed controller of Section 3.3.5 will drive the system to the desired state \mathbf{q}_t^* . The interaction forces can be adapted in our approach, if it is physically possible, using conditioning in a similar fashion. To this end, the user has to specify the desired force or torque, i.e., \mathbf{s}_t^* and accuracy $\boldsymbol{\Sigma}_s^*$. We evaluate the conditioning operator in Section 3.4.1 for conditioning the to desired positions and in Section 3.4.3.

3.3.5 Robot Control with Model-Free ProMPs

We derive a stochastic feedback controller which is ideally capable of reproducing the learned distribution. We define as $\tilde{\mathbf{y}}_t$ the observable state of the system, that contains

the joint positions, velocities, and potentially force or torque data, but not the action. We rewrite the joint probability

$$p(\mathbf{y}_t) = p(\tilde{\mathbf{y}}_t, \mathbf{u}_t) = \mathcal{N} \left(\begin{bmatrix} \tilde{\mathbf{y}}_t \\ \mathbf{u}_t \end{bmatrix} \middle| \tilde{\Phi}_t \boldsymbol{\mu}_w, \tilde{\Phi}_t \boldsymbol{\Sigma}_w \tilde{\Phi}_t^T + \boldsymbol{\Sigma}_y \right),$$

where

$$\tilde{\Phi}_t \boldsymbol{\Sigma}_w \tilde{\Phi}_t^T = \begin{bmatrix} \Phi_t \boldsymbol{\Sigma}_w \Phi_t^T & \Phi_t \boldsymbol{\Sigma}_w \Psi_t^T \\ \Psi_t \boldsymbol{\Sigma}_w \Phi_t^T & \Psi_t \boldsymbol{\Sigma}_w \Psi_t^T \end{bmatrix}, \quad (3.18)$$

and condition on the current observable state $\tilde{\mathbf{y}}_t$ to obtain the desired action. From the Bayes theorem, we obtain the probability of the desired action

$$p(\mathbf{u}_t | \tilde{\mathbf{y}}_t) = \frac{p(\tilde{\mathbf{y}}_t, \mathbf{u}_t)}{p(\tilde{\mathbf{y}}_t)} = \mathcal{N}(\mathbf{u}_t | \boldsymbol{\mu}_u, \boldsymbol{\Sigma}_u), \quad (3.19)$$

which is a Gaussian distribution as both $p(\tilde{\mathbf{y}}_t)$ and $p(\mathbf{u}_t)$ are Gaussian. The mean and covariance of $p(\mathbf{u}_t)$ are computed by

$$\boldsymbol{\mu}_u = \Psi_t \boldsymbol{\mu}_w + \mathbf{K}_t (\tilde{\mathbf{y}}_t - \Phi_t \boldsymbol{\mu}_w) \quad (3.20)$$

$$\boldsymbol{\Sigma}_u = \Psi_t \boldsymbol{\Sigma}_w \Psi_t^T + \mathbf{K}_t \Phi_t \boldsymbol{\Sigma}_w \Phi_t^T, \quad (3.21)$$

$$\mathbf{K}_t = \Psi_t \boldsymbol{\Sigma}_w \Phi_t^T (\Phi_t \boldsymbol{\Sigma}_w \Phi_t^T)^{-1}, \quad (3.22)$$

using Gaussian identities. We rewrite the mean control given the observable state $\tilde{\mathbf{y}}_t$ as

$$\boldsymbol{\mu}_u = \Psi_t \boldsymbol{\mu}_w + \mathbf{K}_t \tilde{\mathbf{y}}_t - \mathbf{K}_t \Phi_t \boldsymbol{\mu}_w = \mathbf{K}_t \tilde{\mathbf{y}}_t + \mathbf{k}_t,$$

and observe that it has the same structure as a feedback controller with time varying gains. The feedback gain matrix \mathbf{K}_t couples the DoF and the additional force-torque signals of the system. The control covariance matrix $\boldsymbol{\Sigma}_u$ introduces correlated noise in the controls. The noise used only if we want to match the variability of the demonstrations. Alternatively, we can disable the noise and replay the noise-free behavior.

3.3.6 Correction Terms for Non-Linear Systems

A basic assumption for the linear feedback controller obtained by the ProMP approach is that the movement is defined in a local vicinity such that a linear controller is sufficient. Whenever the robot's state "leaves" this vicinity, due to the non-linearities of the dynamics, the learned feedback controller might not be able to direct the robot back to the desired trajectory distribution. Therefore, we apply a correction controller that is active only when the state is sufficiently "far" outside the distribution and

directs the system to the mean of the demonstrated state distribution. The correction controller is defined as a standard PD controller with hand-tuned gains, i.e.,

$$\mathbf{u}_t^C = \mathbf{K}_P (\boldsymbol{\mu}_{q,t} - \mathbf{q}_t) + \mathbf{K}_D (\boldsymbol{\mu}_{\dot{q},t} - \dot{\mathbf{q}}) + \mathbf{u}_{\text{ff},t}, \quad (3.23)$$

where the feed forward term $\mathbf{u}_{\text{ff},t}$ is still estimated from the ProMP and given by the mean action of the ProMP for time step t , i.e.,

$$\mathbf{u}_{\text{ff},t} = \mathbf{K}_t \Phi_t \boldsymbol{\mu}_w + \mathbf{k}_t. \quad (3.24)$$

The correcting action \mathbf{u}_t^C is only applied if we are outside the given trajectory distribution. We use a sigmoid activation function that depends on the log-likelihood of the current state to switch between the ProMP feedback controller and the correction controller,

$$\sigma(\mathbf{q}_t, \dot{\mathbf{q}}_t) = \frac{1}{1 + \exp(-\log(p(\mathbf{q}_t, \dot{\mathbf{q}}_t; \boldsymbol{\theta})) \beta^{-1} - \alpha)}, \quad (3.25)$$

where α and β are hand tuned parameters of the activation function. We linearly interpolate between the controls of the ProMP and the correction action. For a high likelihood, e.g., $\sigma(\mathbf{q}_t, \dot{\mathbf{q}}_t) = 1$ we fully activate the feedback controller from the ProMP. For $\sigma(\mathbf{q}_t, \dot{\mathbf{q}}_t) = 0$ we fully activate the correction action.

3.3.7 Time adaptation and mixture of primitives

In this section, we extend our approach to naturally combine multiple primitives, necessary in tasks that the interaction with an object does not always occur at a specific point in time, or not occur at all. For example, when pushing a box, the robot would have to modulate its controller if it is in contact with the box. Placing the box in different locations and training our approach as presented in Section 3.3.2, will create a primitive that averages over both cases, contact and no-contact, and the robot will fail to reproduce the task.

Therefore, we introduce a primitive mixture model that allow the robot to automatically select the best primitive to execute, according to the sensory input and its position. Solely selecting a primitive would have limited use without being able to locally adjust the time to match, for example, exactly the time of contact with the object.

To compensate for time offsets in the movement, we substitute the explicit time relationship in our representation, with a function of time

$$z(t) = t + b, \quad (3.26)$$

which we define as phase. To incorporated the phase variable in Equation (3.3), we modify the basis functions Φ_t to depend on the phase instead of explicitly in time.

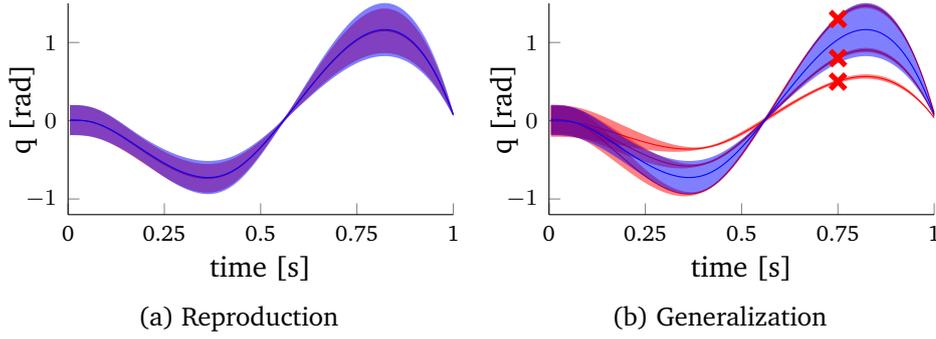


Figure 3.1: (a) We evaluate our approach on a simulated 1-DoF linear system. We use $N = 30$ demonstrations (red) for training. During the reproduction (blue) our approach matches exactly the demonstrations. (b) We evaluate the generalization capabilities of our approach with *conditioning*. The initial distribution is depicted in blue. At time $t = 0.75s$ we condition the initial distribution to pass at a specific position $q = \{0.5, 0.8, 1.3\}$ with low variance. We generate $N = 30$ demonstrations for every conditioning point and we show the resulting distribution in red. The X markers denotes the position at the conditioning point.

Since the phase is a function of time we keep the same notation for clearness. During learning, we estimate a probability distribution over the offset parameter b ,

$$p(b; \boldsymbol{\theta}_b) \sim \mathcal{N}(b | \boldsymbol{\mu}_b, \boldsymbol{\Sigma}_b) \quad (3.27)$$

by fitting a Gaussian distribution. The probability distribution over the state of the system \mathbf{y}_t is now given by

$$p(\mathbf{y}_t; \boldsymbol{\theta}, \boldsymbol{\theta}_b) = \int p(\mathbf{y}_t | b; \boldsymbol{\theta}) p(b; \boldsymbol{\theta}_b) db \approx \sum_{i=1}^L p(\mathbf{y}_t | b_i; \boldsymbol{\theta}), \quad (3.28)$$

which is approximated with L samples as it can not be computed analytically. The samples are being drawn from the prior $p(b; \boldsymbol{\theta}_b)$. The probability of the state given the weight parameters and the sampled offset b_i , $p(\mathbf{y}_t | b_i; \boldsymbol{\theta})$, can be computed in closed form from Equation (3.8).

During reproduction, we select the primitive and offset sample that result in the highest likelihood, i.e.,

$$b^*, \boldsymbol{\theta}^* \approx \arg \max_{b, \boldsymbol{\theta}} p(\mathbf{y}_t | b_i; \boldsymbol{\theta}_j), \quad (3.29)$$

where $\boldsymbol{\theta}^*$ denote the parameters of the most suitable primitive and b^* the time offset.

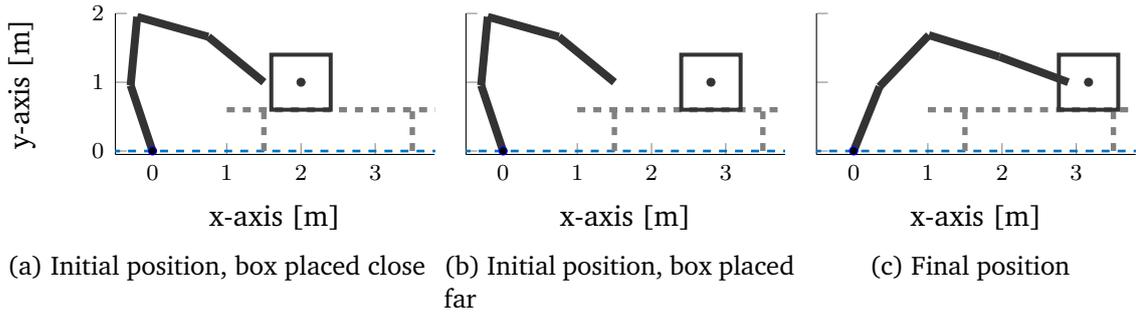


Figure 3.2: The setup of the box the generalization to different initial positions. We use a quad-link, joint control robot with one meter links to push with its end-effector a box to the final configuration, shown in (c). We demonstrated two primitives with the box placed to different initial positions, one within the proximity of the robot’s end-effector (a) and one where the box was placed further away (b).

3.4 Experimental Evaluation

We begin the experimental evaluation on illustrative examples to demonstrate the properties of our approach. We start with a linear one dimensional system to demonstrate the accurate reproduction and generalization of the learned trajectory distribution and we proceed by applying our approach to a more complex system, a quad-link pendulum with non-linear dynamics.

Further, we perform evaluations on complex real-robot platforms. The humanoid robot iCub lifts a grate to a predetermined height from different grasping locations, without learning a model of the grate. Subsequently, we evaluate our approach on moving a chemistry flask of an unknown weight to a target location, while avoiding obstacles, using the KUKA LWR robotic arm.

3.4.1 Reproduction and Generalization of the Trajectory Distribution

In this section, we evaluate the approach on learning a trajectory distribution from demonstrations, generalizing the distribution to novel locations, and reproducing the learned distribution using our proposed control approach. For this evaluation, we used an one dimensional, linear, system with second order integrator dynamics. The demonstrations, used for illustrative purposes, were generated using fifth order splines to reach different via-points, followed by PD control law. We injected noise in the acceleration of the system. The resulting trajectory distribution is shown in Fig. 3.1a (red). In the same figure, we illustrate the resulting trajectory distribution by using our proposed control approach in blue. Our proposed controller matches the demonstrated distribution accurately.

Further, the adaptation capabilities of our approach are evaluated using the conditioning operation that adapts the trajectory distribution to novel situations. We conditioned the trajectory distribution to reach positions 0.5, 0.8 and 1.3 at time point $t = 0.75s$. Our proposed control approach manages to reach the desired position on

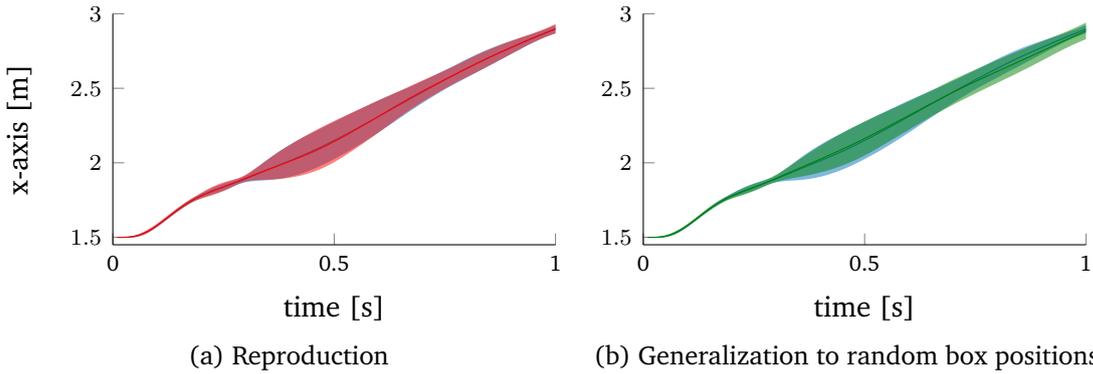


Figure 3.3: The end-effector trajectory for pushing a box placed randomly on the table. In (a), the robot can reproduce exactly the demonstrations. In (b), we demonstrate that the robot reproduces the same end-effector trajectory, for any position between the two demonstrations.

every case as shown in Fig. 3.1b. The desired positions are indicated by red crosses. Our approach maintains the shape of the distribution and reaches the desired position with minimal deviations.

3.4.2 Generalization to different initial positions using a mixture of primitives

In this section, we present an evaluation of our approach using the proposed mixture of primitives to accommodate accurate reproduction when the initial position, and, hence, the time the end-effector of the robot makes contact with the object is unknown. In this demonstration, we used a quad link robot pushing a box placed on top of a table. The position of the box varies and is not observable from our approach. We trained our probabilistic model using three sets of twenty demonstrations, one set were the box’s initial position on the x-axis was set to 2m, a set were the box was placed at 3m, and a primitive where there was no box present. In all three primitives the end-effector was following the same trajectory distribution. The demonstrations were created using a hand-tuned controller. In this evaluation, the robot is joint controlled for both the demonstrations and the reproduction. We simulate the interaction between the robot’s end-effector and the box using a compliant spring-damper model. Hence, the robot’s end-effector can slightly penetrate the box. The experimental setup is depicted in Fig. 3.2.

First, we evaluate our approach on the same scenarios as the demonstrations. We present our results for all three primitives in Fig.3.3a. The robot’s end-effector using our proposed control approach follows the same trajectory distribution (red) as the demonstrations (blue). Additionally, we evaluate our approach using random initial positions for the box in the range 1m to 2m. The robot can reproduce successfully the movement for all the positions in this range. The resulting end-effector trajectory distribution for twenty reproductions is presented in Fig. 3.3b.

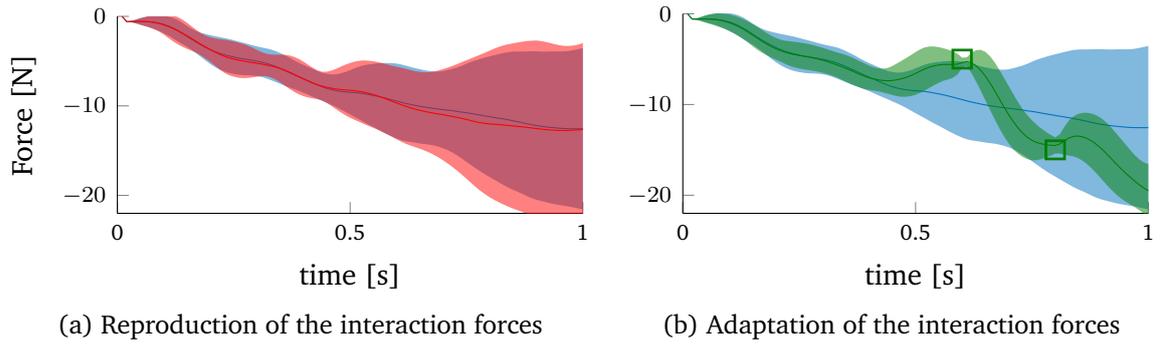


Figure 3.4: In this figure, we show the interaction forces between the robot’s end-effector and a heavy box. In (a), the robot exerts the same force profile (red) as in the demonstrations (blue). In (b), the robot reproduces the adapted force profile (green). The green boxes illustrate the adaptation points.

3.4.3 Adaptation of the interaction forces

In this section, we evaluate our approach on adapting the interaction forces during the execution of the learned primitive, to the desired values. We use the same setup as in Sec. 3.4.2, however, for this evaluation we set the mass of the box high enough for the robot not to be able to push it. We generated demonstration using a hand-tuned controller and we used our approach to reproduce the learned primitive. The robot replicates the same interaction force distribution as observed during the demonstrations. We present our results in Fig. 3.4a.

To evaluate the adaptation capabilities of the interaction forces, we conditioned the learned primitive to apply -5N at time $t = 0.6\text{s}$ and -15N at $t = 0.8\text{s}$. During the execution of the primitive, the robot successfully reproduces the desired forces at the corresponding time, while during for the remaining time the interaction forces generated using the proposed controller were close to the demonstrations. We show our results in Fig. 3.4b.

3.4.4 Non-Linear Quad-Link Pendulum

To evaluate the quality of our controller on a non-linear system, we tested our model-free ProMP approach on a non-linear quad-link planar pendulum. Each link had a mass of 1kg and a length of 1m . We used the standard rigid body dynamics equations, where the gravity and the Coriolis forces are the major non-linear terms. We collected demonstrations by defining the desired trajectory as a spline with two via-point at $t = 0.3, 0.8$ in the task-space of the robot. We generated the demonstration trajectories using inverse kinematics for generating the joint space reference trajectories. Then, we used a inverse dynamics controller to track the reference trajectories and we collected the joint state-action data. We trained our approach using $N = 30$ demonstrations.

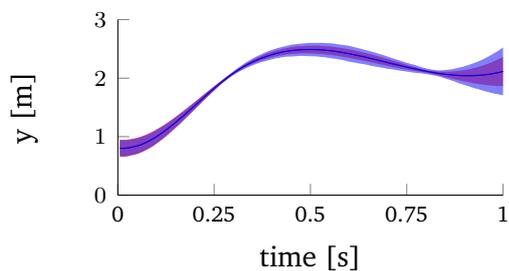


Figure 3.5: We evaluate our approach on a non-linear system with $D = 4$ DoF. While the dynamics of the task are non-linear we are able to reproduce (blue) accurately the demonstrated distribution (red). We show the trajectory distribution of the “y” dimension of the task-space of the robot. Our approach captures the correlations between the DoF of the robot and reduces the variance of the trajectory reproduction at both via-points.

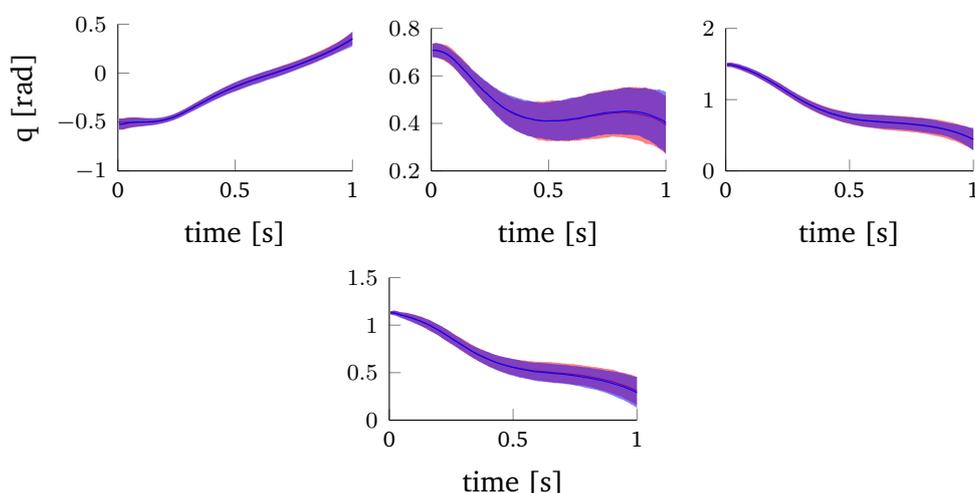


Figure 3.6: The evaluation of our approach on the quad-link robot. We present the results of the DoF in joint space. The demonstrated distribution is plotted in red and the reproduction in blue. The two distributions match. The two via-points of the movement, which were set in task-space, are not visible in joint-space.

The resulting trajectory distribution for the y-dimension of the task-space is shown in Fig. 3.5. The robot can track with its end-effector the desired distribution accurately and can reproduce the two via-points. In Fig. 3.6 we show all four joint trajectories. In the joint space distributions the via-points are not visible but are captured in the covariance matrix of the weights. While the distribution is wide, the controller could match the mean and variance of the demonstrated trajectory distribution. In Fig. 3.7, we illustrated the resulting trajectory from the controller in the task space of the robot. The activation of the correcting controller is around 1% of the total execution time.

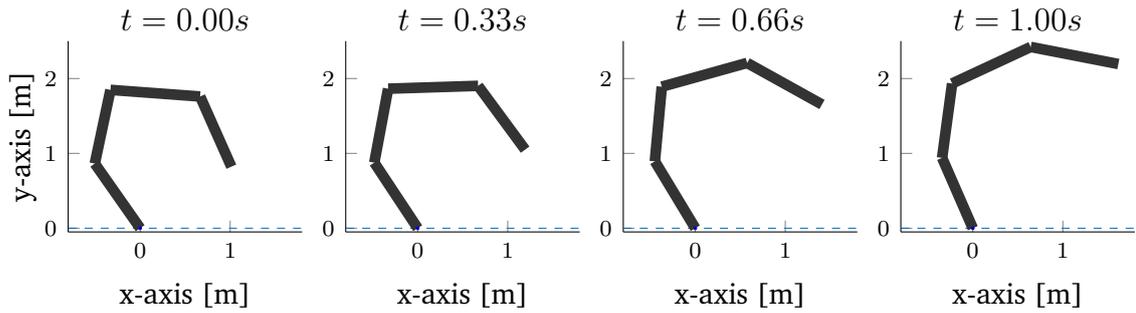


Figure 3.7: An animation of the movement of the quad-link non-linear robot during the execution of our approach. We use darker colors at the beginning of the movement and lighter at the end.

3.4.5 Adaptation to External Forces on the iCub

In this experiment we used the presented model-free ProMP approach to learn a one-dimensional torque feedback controller in the humanoid robot iCub. The task is to tilt a grate multiple times from an initial distribution to a goal distribution, as shown in Fig. 3.9a. In our experiments we use the wrist joint. The grate is attached to the robot at different lengths, to simulate different grasping locations. We demonstrate 20 movements per grasping location to train our approach. The data were recorded through tele-operation. In this experiment the state encodes the joint angle encoder value and the joint torque reading in the wrist. We present the recorded torques from the sensor of the robot for all three demonstrated grasping locations in Fig. 3.9b. By placing the grate on the same location as during the demonstration and reproducing the movement with our approach, we show that we observe the same torque profile. The force measurement is crucial in our experiment as it is used for applying the correct forces during the execution of the movement. When disabled, the robot either fails to lift the grate to the demonstrated location or it overshoots. The overshooting is due to gravity, as in that grasping location the center of the mass of the grate is moved over the axis of wrist rotation. The results are shown in Fig. 3.10. The reproduction distributions were created using twenty executions of the model-free ProMP controller per grasping location. Our approach can generalize to different grasping locations between the demonstrations. We generalized into four new locations and executed our controller. The robot reproduces the same joint distribution while compensating for the different dynamics, as shown in Fig. 3.9a.

3.4.6 Repositioning a chemistry flask

In this experiment we evaluate our approach on repositioning a chemistry flask filled with an unknown amount of liquid. The flask can not be moved in a straight line to the end position as another flask occludes the path. Rather, the robot has to follow a curved trajectory to the end point. We used the KUKA LWR robot for the experiment. The dimensionality of state-action distribution is seventeen, seven for the degrees of freedom of the robot, three for the Cartesian forces, and seven for the controller ac-

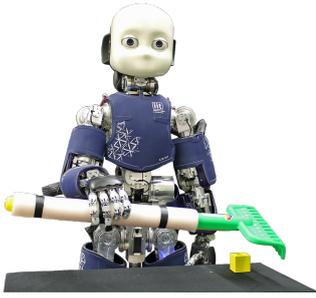


Figure 3.8: The *iCub* robot is taught by imitation how to tilt a grate that we use of during the experimental evaluation of our approach. We demonstrated how to lift a grate from three different positions. Grasping from different positions change the dynamics of the task. Our method provides online adaptation and generalizes in the area of the grasps

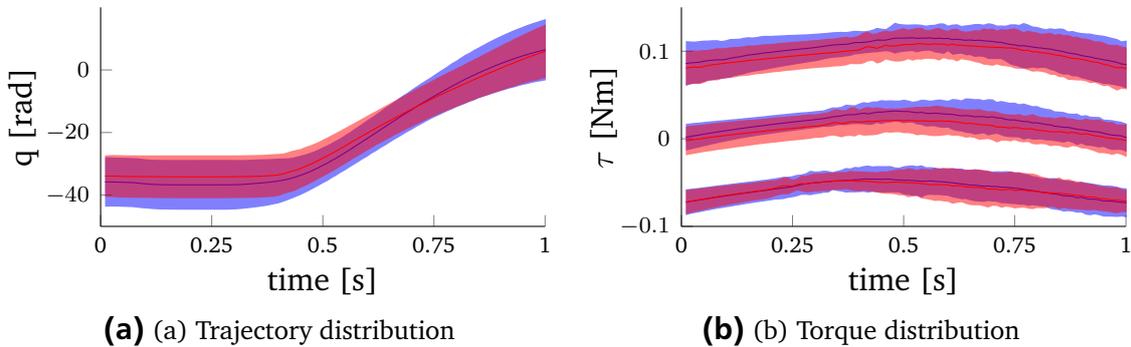


Figure 3.9: (a) The trajectory distribution of the wrist joint of the *iCub* during our experiment. The demonstrated distribution is presented in blue and the reproduction in red. The demonstrated distribution contain trajectories from all three grasping locations. The reproduction distribution contain trajectories from seven grasping locations. The Model-Free ProMPs can reproduce the demonstrated distribution in new grasping locations. (b) The torque distribution of all grasping locations used during the demonstrations. Each location created a distinct offset in the measured torque. We present the demonstrated torque distributions in blue. Additionally, we show that our approach can reproduce the torque distribution when we position the grate at the same locations as in the demonstrations. We present the reproduction results in red.

tions. We presented the robot with two sets of ten demonstrations for two different amounts of liquid, 200ml and 400ml. Using the demonstrations, we trained a primitive that encodes the shape of the movement, the corresponding force data, and the observed actions. The interaction between the flask, the robot’s end-effector, and the table cloth is not directly modelled.

Controlling the robot with the proposed approach, the robot reproduced the learned trajectory distribution for both liquid levels, 200ml and 400ml, without observing the amount of the liquid, as we present in 3.13a. Additionally, the robot reproduced successfully the task with the flask filled at 300ml. The robot reproduces the trajectory distribution of the demonstrations. The interaction forces during the

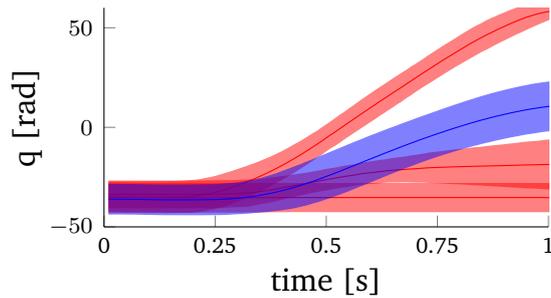


Figure 3.10: The trajectory distribution of the wrist joint of the robot, when we disable the torque feedback. Depending on the grasping location, the robot either fails to lift the grate to the same height as demonstrated, or, it overshoots the lifting task due to gravity. In the later case, it should be noted that the center of mass of the grate is moved over the axis of the joint and, thus, gravity forces the grate to lift. For comparison, we present the demonstration distribution from all grasping locations in blue.

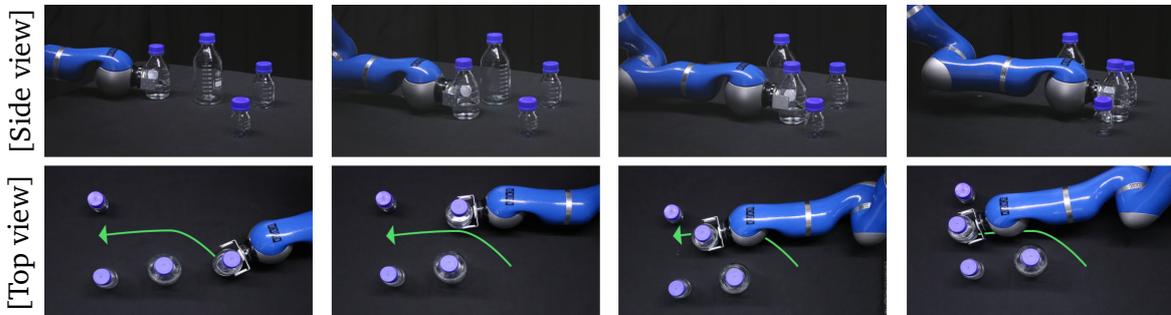


Figure 3.11: Illustration of the experimental setup. The robot repositions a chemistry flask with an unknown amount of liquid, while avoiding obstacles.

execution of the skill are shown in Fig. 3.14. The robot reproduces the similar interaction forces as the demonstrations. In the generalized case of filling the flask at 300ml, the interaction forces are in between the two extremes. To demonstrated the adaptation capabilities of our approach, we used conditioning to move the flask at a novel position at the end of the movement, as shown in Fig. 3.12, for all three liquid levels, 200ml, 300ml, and 400ml. The robot successfully reproduced the task. The end-effector trajectories are shown in Fig. 3.13b.

3.5 Epilogue

In this chapter, we have presented a model-free approach for Probabilistic Movement Primitives (ProMP) that can be used for learning skills for physical interaction with the environment from demonstrations. Our approach neither requires a known model of the system dynamics nor attempts to explicitly train one. Rather, we correlate the actions present during the demonstrations to the state of the robot. We showed how our approach could adapt to changes in the environment, as for example adding via-points, or placing the object to different initial positions. We showed that the model-free ProMP approach inherits many beneficial properties of MPs such as repro-

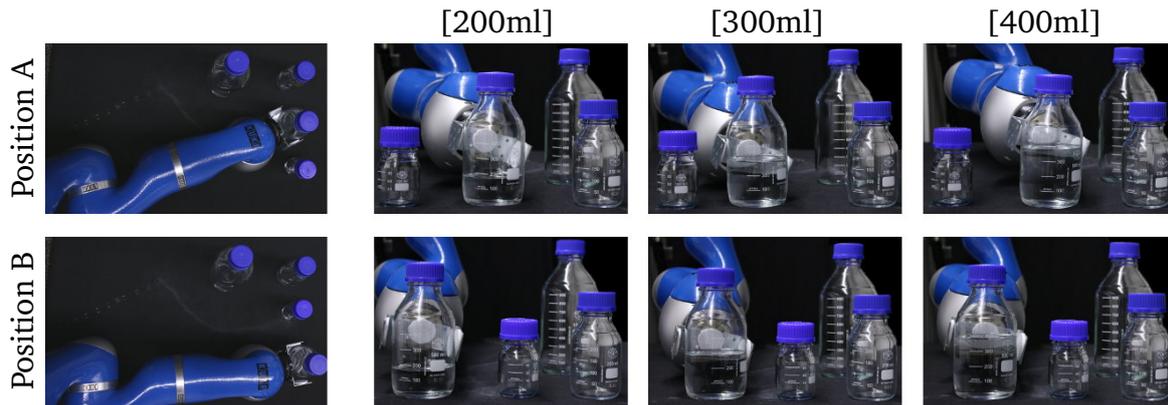


Figure 3.12: Reproduction and generalization of the repositioning skill. We present the configuration of the robot at the end of the movement. The robot was trained for “Position A” and with 200ml and 400ml in the flask. Using our approach, the robot can successfully reproduce the movements and generalize to a different liquid amount, 300ml, and to a different final location, “Position B”.

ducing the variability in the demonstrations as well as using probabilistic operations such as conditioning for generalization to different via points. Our approach is different from directly encoding the actions, as generates the action through a model that depends on the state and the time. Hence, our approach can generalize well in the vicinity of the demonstrations. We derived a stochastic feedback controller that is obtained from the distribution over the trajectories and accurately tracks the demonstrated distribution. Our approach is best suited in tasks where time is critical for the execution of the task, e.g. pushing a button at a specific movement, or grasping a moving object.

For learning physical interaction tasks, we showed that we can include sensory signals, for example the measure torques, in our distribution. By learning the correlations of this sensory signal, we can coordinate the controls needed for the physical interaction with the measured torques and forces. Such coordination is essential for the complex interaction tasks.

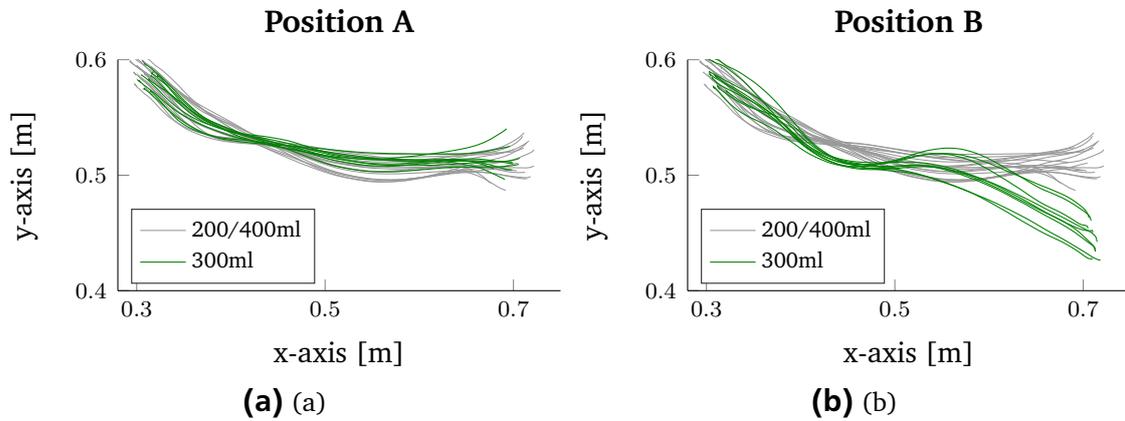


Figure 3.13: The trajectories of the end-effector of the robot at the flask repositioning experiment. We present the training data for both liquid amount, 200ml and 400ml in gray. The reproduction for the 300ml is shown in green. In (a), the robot reproduced the movement achieving the same end-effector distribution as in the demonstrations. In (b), we adapted the final location with conditioning.

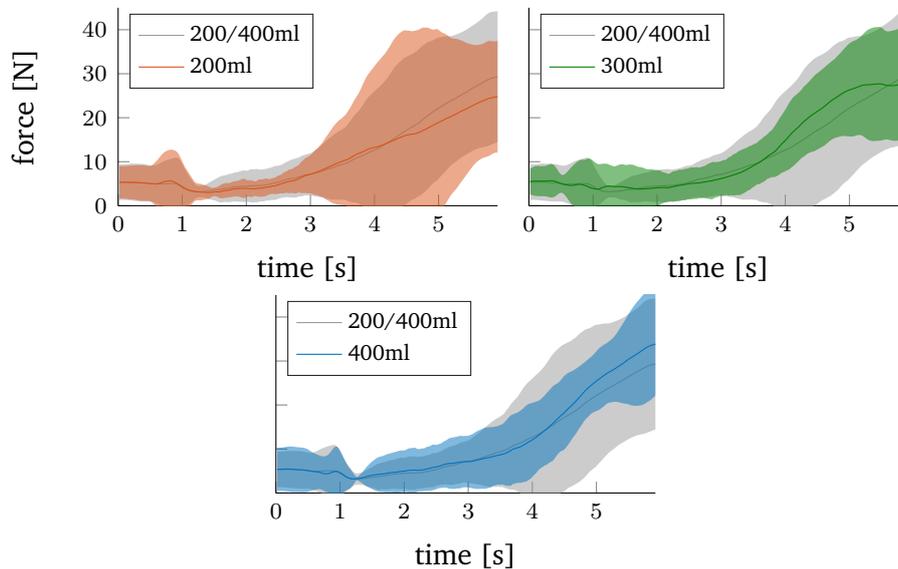
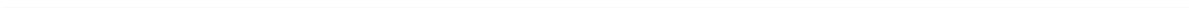


Figure 3.14: We present the force distribution during the execution of the flask repositioning skill. The force profiles exerted during the demonstrations for both liquid amount, 200ml and 400ml, are shown in gray. The reproduction force profiles for both training liquid amounts and the new 300ml level, are within the demonstration distribution.



4 Prioritization of Movement Primitives

4.1 Introduction

Complex robots with redundant degrees of freedom can in principle perform multiple tasks at the same time, as for example, reaching for an object with a humanoid robot while balancing, or reaching for an object with a robotic arm while avoiding an obstacle. However, simultaneously performing multiple tasks using the same degrees of freedom, requires combining the control signals from all the tasks.

While many schemes for combining control signals have been developed, this chapter focuses on approaches that resolve the control combination by prioritizing the tasks, i.e., approaches which assume that one task can be executed with priority over another task, even if the latter will not be performed sufficiently well.

In contrast to current approaches, where the priorities are set by experts, we propose learning the priorities from demonstrations. The Learning from Demonstrations (LfD) paradigm has been very successful in movement generation (Calinon et al., 2010a,b; Calinon, 2016) for complex robotic tasks. In LfD a solution is provided by demonstrations and, therefore, avoids using hand-coded controllers or the setting up a cost function for further planning or optimization. Yet, LfD has not been yet fully explored in prioritized control. LfD can introduce new properties to current prioritization approaches, such as adaptation to new situations and reproduction of unseen combinations of tasks, without retraining.

Further, in many robotic tasks, accurate task reproduction for the whole duration of the task may not affect the task's performance. For example, a successful reproduction of a pick-and-place task, depends on highly accurate movements during picking and placing, but not for the rest of task's execution. In classical approaches, where there is no notion of the task's accuracy, combining tasks results in insufficient performance when the combination of both tasks is not physically possible. However, in our approach, we extract the time-varying task accuracy from demonstrations and we use it to modulate the task prioritization, where the accuracy implicitly defines the task's priority. When a task has low accuracy, our approach allows for deviations from the reference trajectory and, therefore, enables tasks with higher accuracy to be executed at the same time without reaching the physical limitations of the system. Hence, our approach allows for a more efficient combination of tasks.

In this chapter, we propose a novel data-driven framework for learning prioritized task representations, i.e., learn the tasks and the relative priorities of the tasks, from demonstrations. We combine Bayesian task prioritization (Toussaint and Goerick, 2010) that allows the computation of the combined control signal from multiple tasks at different operational spaces with Probabilistic Movement Primitives

(ProMPs) (Paraschos et al., 2013a,b) to learn prioritized complete motion sequences. MPs (Ijspeert et al., 2003; Calinon et al., 2012; Khansari-Zadeh and Billard, 2011; Paraschos et al., 2013a,b), are a powerful representation for encoding complex movements, much more flexible than using attractors or point-to-point movements, and enable adaptation of the learned movements without retraining. No primitive representation has so far taken advantage of introducing task priorities. ProMPs model the task and its desired accuracy from multiple demonstrations, provide a mechanism for adapting a learned task to novel situations, and an acceleration-space control law that follows the encoded task. The demonstrations can be acquired by several imitation learning techniques, including kinesthetic teach-in and tele-operation. We extend the Bayesian task prioritization (Toussaint and Goerick, 2010) to provide a more general derivation for torque control and show that existing prioritization techniques are a special case of the Bayesian approach. We use the ProMP approach as it can naturally be combined with Bayesian task prioritization in a single, principled, probabilistic framework. We derive our approach based on ProMPs, however, other stochastic movement representations could be used analogously (Calinon et al., 2012; Rozo et al., 2013).

We use our approach to learn multiple primitives for different operational spaces, e.g., the end-effector or the center of mass space. Each primitive solves a specific task in the corresponding space. We present how to adapt the task combination to new situations, e.g., reach a different via-point with an end-effector, that can be achieved without explicitly solving an inverse kinematics problem. Furthermore, we demonstrate how multiple primitives of different operational-spaces can now be seamlessly combined in order to achieve a new, unseen combination of tasks. Our prioritized LfD approach can be efficiently used without requiring re-learning the unseen combinations. As shown in our experiments, using prioritization also allows to adapt a library of primitives to changes in the environment, e.g., the introduction of an obstacle in the scene. We use simulations, the humanoid robot “iCub”, and the KUKA LWR robot arm platform to evaluate our approach.

4.2 Related work

A common approach for combining different control signals is to prioritize the tasks, under the assumption that this prioritization is not allowed to be violated. We refer to such schemes as strict prioritization schemes. In these schemes, a higher priority task does not get disturbed by the control signals of the lower priority ones (Peters et al., 2007; Khatib, 1987; Khatib et al., 2004; Nakamura et al., 1987; Sentis and Khatib, 2005; Park et al., 2002; Bruyninckx and Khatib, 2000; Luh et al., 1980; Kober and Peters, 2014). A lower priority task is always projected in the null-space of the high priority task. Although these approaches provide guarantees on the prioritization performance, strict prioritization approaches can get numerically unstable when the robot enters a singular kinematic configuration. Numerical regularization can be used, but it violates the null-space projection (Baerlocher and Boulic, 1998). The introduction of regularization has the side-effect that a low priority task can interfere with a higher priority task. We show how our approach can obtain similar regularizations from demonstrations.

For some tasks, such a prioritization scheme is natural, for example, a humanoid robot should not tip over and, therefore, the balancing controller should always have the highest priority. However, defining a strict priority can be problematic in general. For example, for reaching an object with one hand of a humanoid robot, while simultaneously reaching for a different object with the other hand, it is not clear these tasks can be prioritized. Both tasks could have the same importance, i.e. priority, or, the importance of each tasks could vary in time and depending on, e.g., the desired execution accuracy at that point in time. For such scenarios, the relative importance between the tasks can be easier to set. These problems are partially addressed in (Salini et al., 2011, 2013; Decre et al., 2009), where a “soft” prioritization scheme was introduced. In our approach, we step further and propose to learn the relative priorities from data and, therefore, minimize the amount of parameters that require tuning through expert knowledge.

“Soft” prioritization approaches do not assume a hierarchy of tasks a priori, but use the relative priorities between the tasks. In this scheme, every task contributes to the control signal. The degree of contribution depends on its relative priority. “Soft” prioritization approaches demonstrate promising results (Salini et al., 2011; Lober et al., 2014, 2015) to successfully perform multiple tasks due to the relaxation of the initial problem. Intuitively, “soft” prioritization schemes could be thought of as violating the hierarchy of priorities. They are often stated as multi-objective optimization problems (Salini et al., 2011; Lober et al., 2014, 2015). Each task is formulated as a quadratic cost function and uses the relative priority as weight. The result of the optimization yields the controls that minimize the total cost and, therefore, allow lower priority tasks to perturb higher priority ones as long as the total cost is decreased.

Current prioritization approaches often assume a static prioritization or weighting scheme, where the importance of each task remains constant during the execution of the movement (Khatib et al., 2004; Decre et al., 2009). However, modulating the importance of the tasks during the movement can be beneficial. First, tasks that are no longer desired to be executed can be faded-out and new tasks can be smoothly introduced, without torque jumps. Salini et al. (Salini et al., 2011) proposed to dynamically adjust the priorities for achieving movement sequencing and task transitions. More importantly, the modulation of the priorities can be related to the desired accuracy of the task. During the time-steps with low task-priority, the robot can focus on executing other tasks. Therefore, setting the relative priorities can be a simpler problem than specifying the strict task hierarchy, as the expert has to specify only the time points that require higher accuracy. Lober et al. (Lober et al., 2015) demonstrated that this approach increases the flexibility of the system and decreases “lock-ups” where a more important movement prohibits the execution of less important tasks, while requiring less expert knowledge. Modugno et al. (Modugno et al., 2016) proposed the use of an optimization algorithm to find suitable relative priorities that further decreases expert knowledge.

4.3 Probabilistic Prioritization of Movement Primitives

This chapter presents a generic probabilistic framework for simultaneously combining multiple tasks. We assume that each task has a different level of accuracy and that the

accuracy can change over the execution of the task. The task accuracy is associated with the respective importance for the task combination.

First, we encode the time-varying accuracy in an efficient representation and, importantly, obtain it from imitation data. Second, we develop our stochastic combination approach using the task accuracy as relative priorities. Furthermore, we show that current prioritization approaches can be derived within our probabilistic approach, when some uncertainty parameters are set to zero. Finally, we present an extension to the ProMPs controller that increases the tracking performance when prioritizing several primitives. We extend our approach to multiple operational-space controllers in Sec. 4.3.3. In Sec. 4.4 we show that our stochastic prioritizing scheme can be generalized to a wider class of controllers.

4.3.1 Encoding Task Accuracy from Demonstrations

Representing the desired task accuracy throughout the duration of the task is critical for relative prioritization schemes. A measure for the task accuracy is the task variance that is obtained over multiple executions of the task. Stochastic movement primitive representations can not only represent the task variance but also be trained from demonstration data. To this end, we use the Probabilistic Movement Primitives (ProMPs) approach (Paraschos et al., 2013a,b) as our representation.

ProMPs represent a single trajectory as a weighed linear combination of Gaussian basis functions Φ_t and the respective weights \mathbf{w} , i.e., $\mathbf{y}_t = \Phi_t \mathbf{w}$, where $\mathbf{y}_t = [\mathbf{x}, \dot{\mathbf{x}}]^T$ represents the state of the task, i.e., positions and velocities, at time t . The task state \mathbf{y}_t is a vector that contains the variables that define the state of the tasks, e.g., the joint or end-effector positions and velocities. Each task demonstration is used to estimate the weights \mathbf{w} for that execution using a maximum likelihood approach (Paraschos et al., 2013a). From the set of estimated weights, ProMPs estimate a distribution over the weights, i.e.,

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w} | \boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w), \quad (4.1)$$

which is assumed to be approximated well by a Gaussian, or a Gaussian mixture (Ewerton et al., 2015b; Rueckert et al., 2015b). Thus, ProMPs offer a compact representation of the trajectory distribution in task space, that is, the mean movement, the correlation between the task's variables, and their variance. With ProMPs, we can evaluate the distribution of the state $p(\mathbf{y}_t)$ at every time-step

$$p(\mathbf{y}_t) = \int p(\mathbf{y}_t | \mathbf{w}) p(\mathbf{w}) d\mathbf{w} = \mathcal{N}(\mathbf{y}_t | \boldsymbol{\mu}_{y_t}, \boldsymbol{\Sigma}_{y_t}) \quad (4.2)$$

in closed form. ProMPs also provide a stochastic linear feedback controller, which is also derived in closed form. The controller can follow the encoded task distribution exactly, i.e., it matches mean and variance of the distribution. In (Paraschos et al., 2013a,b), ProMPs are used to control the joints of the robot and, therefore, the controller outputs are joint torques. In this chapter, we generalize ProMPs to model and control in operational space, e.g., the robot's end-effector space. To do so, we ad-

just the ProMP controller’s output to be the acceleration of task space variables. The stochastic controller is, therefore, given by

$$p(\ddot{\mathbf{x}}|\mathbf{y}_t) = \mathcal{N}(\ddot{\mathbf{x}}|\mathbf{K}_t\mathbf{y}_t + \mathbf{k}_t, \Sigma_{\ddot{\mathbf{x}}}). \quad (4.3)$$

The mean of the controller is given by a linear feedback control law. The controller additionally contains the covariance of the task in the acceleration space, which plays an important part in our approach as it specifies the required accuracy of the control, see Sec. 4.3.2. In summary, ProMPs are capable of representing and learning the task covariance Σ_y , and transforming it to the acceleration covariance $\Sigma_{\ddot{\mathbf{x}}}$.

4.3.2 Probabilistic Combination of Tasks

We begin our derivation given two tasks, a joint-space task and an operational-space task. For each task, a stochastic controller is obtained from the corresponding ProMP that has been trained from demonstrations. Every output of each controller is normally distributed, i.e.,

$$p_1(\ddot{\mathbf{q}}) \sim \mathcal{N}(\ddot{\mathbf{q}}|\boldsymbol{\mu}_{\ddot{\mathbf{q}}}, \Sigma_{\ddot{\mathbf{q}}}), \quad p_2(\ddot{\mathbf{x}}) \sim \mathcal{N}(\ddot{\mathbf{x}}|\boldsymbol{\mu}_{\ddot{\mathbf{x}}}, \Sigma_{\ddot{\mathbf{x}}}). \quad (4.4)$$

The vector $\ddot{\mathbf{q}}$ denotes the joint acceleration for all of the joints of the robot and the vector $\ddot{\mathbf{x}}$ the operational-space acceleration. We drop the time-index for simplicity.

The operational-space controller and the joint-space controller can not be used simultaneously without accounting for the kinematics of the system. The system kinematics introduce a constraint between the operational and the joint space acceleration. The constraint is commonly defined in the velocity space by $\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}$, where \mathbf{J} denotes the Jacobian from a base-frame to the operational-space. Equivalently, by differentiation over time, we obtain the acceleration-space formulation $\ddot{\mathbf{x}} = \mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}}$ of the constraint, where $\dot{\mathbf{J}}$ denotes the time derivative of the Jacobian. Given the constraint in the acceleration-space, the operational-space controller depends on the current joint-acceleration $\ddot{\mathbf{q}}$. The probability of the operational-space acceleration $\ddot{\mathbf{x}}$ given the joint acceleration $\ddot{\mathbf{q}}$ is defined as the conditional

$$p_{2|\ddot{\mathbf{q}}}(\ddot{\mathbf{x}}|\ddot{\mathbf{q}}) \sim \mathcal{N}(\ddot{\mathbf{x}}|\mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}}, \Sigma_{\ddot{\mathbf{x}}}), \quad (4.5)$$

where the mean of the conditional distribution is given by the constraint and the variance is given by the desired task accuracy. We can now use the joint space ProMP as prior distribution and the desired task-space mapping $p_{2|\ddot{\mathbf{q}}}(\ddot{\mathbf{x}} = \boldsymbol{\mu}_{\ddot{\mathbf{x}}}|\ddot{\mathbf{q}})$ as likelihood to obtain the posterior distribution for the joint space controller using Bayes theorem, i.e.,

$$p_{1|\ddot{\mathbf{x}}}(\ddot{\mathbf{q}}|\ddot{\mathbf{x}} = \boldsymbol{\mu}_{\ddot{\mathbf{x}}}) = \frac{p_{2|\ddot{\mathbf{q}}}(\ddot{\mathbf{x}} = \boldsymbol{\mu}_{\ddot{\mathbf{x}}}| \ddot{\mathbf{q}})p_1(\ddot{\mathbf{q}})}{p_2(\ddot{\mathbf{x}})} = \mathcal{N}(\ddot{\mathbf{q}}|\boldsymbol{\mu}, \Sigma).$$

The control law for the joint accelerations $\ddot{\mathbf{q}}$ is then obtained by computing the marginal distribution

$$p_{1|2}(\ddot{\mathbf{q}}) = \int p_{1|\ddot{\mathbf{x}}}(\ddot{\mathbf{q}}|\ddot{\mathbf{x}})p_2(\ddot{\mathbf{x}})d\ddot{\mathbf{x}} = \mathcal{N}(\ddot{\mathbf{q}}|\boldsymbol{\mu}'_{\ddot{\mathbf{q}}}, \boldsymbol{\Sigma}'_{\ddot{\mathbf{q}}}). \quad (4.6)$$

The mean $\boldsymbol{\mu}'_{\ddot{\mathbf{q}}}$ and the covariance $\boldsymbol{\Sigma}'_{\ddot{\mathbf{q}}}$ are computed as

$$\boldsymbol{\mu}'_{\ddot{\mathbf{q}}} = \mathbf{J}^\dagger (\boldsymbol{\mu}_{\ddot{\mathbf{x}}} - \mathbf{J}\dot{\mathbf{q}}) + (\mathbf{I} - \mathbf{J}^\dagger \mathbf{J}) \boldsymbol{\mu}_{\ddot{\mathbf{q}}} \quad (4.7)$$

$$\boldsymbol{\Sigma}'_{\ddot{\mathbf{q}}} = (\mathbf{I} - \mathbf{J}^\dagger \mathbf{J}) \boldsymbol{\Sigma}_{\ddot{\mathbf{q}}} + \mathbf{J}^\dagger \boldsymbol{\Sigma}_{\ddot{\mathbf{x}}} \mathbf{J}^{\dagger T} \quad (4.8)$$

where the \mathbf{J}^\dagger denotes the generalized inverse of the Jacobian

$$\mathbf{J}^\dagger = \boldsymbol{\Sigma}_{\ddot{\mathbf{q}}} \mathbf{J}^T (\boldsymbol{\Sigma}_{\ddot{\mathbf{x}}} + \mathbf{J} \boldsymbol{\Sigma}_{\ddot{\mathbf{q}}} \mathbf{J}^T)^{-1}. \quad (4.9)$$

In our approach, the joint space acceleration $\ddot{\mathbf{q}}$ and the task-space acceleration $\ddot{\mathbf{x}}$ are obtained from the stochastic feedback controller of the ProMPs. The variance of the task-space controller $\boldsymbol{\Sigma}_{\ddot{\mathbf{x}}}$ is used as the regularization matrix and the variance of the joint-space controller $\boldsymbol{\Sigma}_{\ddot{\mathbf{q}}}$ as weighting. The regularization matrix $\boldsymbol{\Sigma}_{\ddot{\mathbf{x}}}$ is full-rank.

4.3.3 Extension to Multiple Tasks

Multiple operational-space controllers can be naturally integrated in our approach where each task $i \in 1 \dots N$ can operate in a different operational space. In principle, it is sufficient to compute the posterior distribution over the joint acceleration $\ddot{\mathbf{q}}$, given the accelerations of all task controllers $\{\ddot{\mathbf{x}}_i\}_{1 \dots N}$, i.e., $p(\ddot{\mathbf{q}}|\{\ddot{\mathbf{x}}_i\}_{1 \dots N})$, that can be computed recursively, or in a single step (Toussaint and Goerick, 2010). The single step solution does not relate to existing prioritization approaches.

For the recursive computation, we start with our prior distribution over the joint accelerations $p_1(\ddot{\mathbf{q}})$. We condition it with the operational-space acceleration distribution $p_N(\ddot{\mathbf{x}}_N)$ of the highest priority task. The resulting posterior distribution $p_{1|N}(\ddot{\mathbf{q}}|\ddot{\mathbf{x}}_N)$ is then used as a new prior distribution and is conditioned with $p_{N-1}(\ddot{\mathbf{x}}_{N-1})$. We continue conditioning until we reach the task $i = 1$. During the computation of the new prior distribution, we can perform a numerical stability analysis of the matrix, $\boldsymbol{\Sigma}_{\ddot{\mathbf{x}}} + \mathbf{J} \boldsymbol{\Sigma}_{\ddot{\mathbf{q}}} \mathbf{J}^T$, e.g., by computing the condition number of the matrix. If the inversion becomes numerically unstable, then the task i_o added at this step is incompatible to the already added tasks $N \dots i_o - 1$. The order of inference should be chosen by starting from the most important to the least important tasks, when physical limitations are reached, no more tasks would make sense to be added. Otherwise, the order of inference does not modify the resulting controller in our approach. Our recursive approach has similarities with the hierarchical prioritization approaches presented in (Khatib, 1987; Peters et al., 2007). However, in our approach we use the regularized generalized inverse, as presented in Sec. 4.3.2, where the tasks accuracies are obtained from imitation data, instead of treating each task with an infinite accuracy that can cause numerically unstable solutions.

Table 4.1: Comparison of different pseudo inverses used for operations

$J^\dagger = J^T (J J^T)^{-1}$	Generalized inverse
$J^\dagger = M^{-1} J^T (J M^{-1} J^T)^{-1}$	Weighted generalized inverse, weighted with the inverse of the mass
$J^\dagger = \Sigma_{\ddot{q}} J^T (\Sigma_{\ddot{x}} + J \Sigma_{\ddot{q}} J^T)^{-1}$	Bayesian inverse, weighting and regularization are computed in closed form.

4.3.4 Robust Trajectory Distribution Tracking

We use the ProMP controller to get the desired accelerations in both, the task and joint spaces. The controller is based on (Paraschos et al., 2013a), where the authors derive Equation (4.3) by matching the change in the sufficient statistics of the system, i.e., change of mean and covariance, at the current time-step. Hence, it is assumed that $\mu_t, \dot{\mu}_t, \Sigma_t$ and $\dot{\Sigma}_t$ are known.

Due to the prioritization of multiple primitives, if deviations that were not present in the demonstrations occur and the system drift away from the demonstrated area. The controller computation presented in (Paraschos et al., 2013a) generates gains that are not optimal for the drifted state distribution and these errors yield in an inaccurate tracking behavior, where the reproduction distribution does not match the demonstrated.

We propose to adjust the current belief of the mean state μ_t and its derivative $\dot{\mu}_t$, according to the current observation of the state \mathbf{y}_t . We adapt the mean belief μ_t as a weighted average of the mean state obtained from demonstrations μ_t^{demo} and the current state \mathbf{y}_t as

$$\mu_t = \gamma \mu_t^{\text{demo}} + (1 - \gamma) \mathbf{y}_t, \quad (4.10)$$

where γ is computed by a sigmoid activation based on the likelihood of the current state, i.e.,

$$\gamma(\mathbf{y}_t) = (1 + \exp(-\log(p(\mathbf{y}_t; \theta)) \beta^{-1} - \alpha))^{-1}, \quad (4.11)$$

where α, β are open parameters. Additionally, we adapt the time derivative of the mean state $\dot{\mu}_t$ with a feedback controller to converge to the demonstrated $\dot{\mu}_t^{\text{demo}}$, i.e.,

$$\dot{\mu}_t = \begin{bmatrix} \dot{\mu}_{\dot{q}_t} \\ \dot{\mu}_{\ddot{q}_t} \end{bmatrix} = \begin{bmatrix} \mu_{\dot{q}_t}^{\text{demo}} \\ K^{\text{SC}} (\mu_t^{\text{demo}} - \mu_t) + \mu_{\dot{q}_t}^{\text{demo}} \end{bmatrix}, \quad (4.12)$$

where K^{SC} are feedback gains. If the current state is inside the distribution, γ will be 1 and the correction term for the mean will not be activated. However, if the current state is outside the distribution, i.e., we have a small likelihood, it is an indication that inaccuracies in the controller computation accumulated, such that the distribution is

not matched any more. In this case, the additional feedback controller is used to push the mean back to the desired mean, and, hence, increasing the tracking performance.

4.4 Relation to Optimization-Based Prioritization Schemes

Our approach can be generalized to a wider class of problems, where the constraints are linear to the joint acceleration $\ddot{\mathbf{q}}$, i.e., can be formulated as $\mathbf{A}\ddot{\mathbf{q}} = \mathbf{b}$, where the matrix \mathbf{A} and vector \mathbf{b} possibly depend on the current state of the robot \mathbf{q} and $\dot{\mathbf{q}}$ at time t . The constraint imposed by the robot's mechanics can be re-formulated in the generalized form by setting $\mathbf{A} = \mathbf{J}$ and $\mathbf{b} = \ddot{\mathbf{x}} - \mathbf{J}\dot{\mathbf{q}}$.

We formulate an optimization problem that incorporates a soft version of this constraint while staying close to the prior mean. The covariance matrices serve as L2 norm metrics for the objectives, i.e.,

$$\begin{aligned} \arg \min_{\ddot{\mathbf{q}}} J = \arg \max_{\ddot{\mathbf{q}}} & (\mathbf{A}\ddot{\mathbf{q}} - \mathbf{b})^T \Sigma_{\ddot{\mathbf{x}}}^{-1} (\mathbf{A}\ddot{\mathbf{q}} - \mathbf{b}) \\ & + (\ddot{\mathbf{q}} - \boldsymbol{\mu}_{\ddot{\mathbf{q}}})^T \Sigma_{\ddot{\mathbf{q}}}^{-1} (\ddot{\mathbf{q}} - \boldsymbol{\mu}_{\ddot{\mathbf{q}}}). \end{aligned} \quad (4.13)$$

This formulation resembles the optimization framework presented in (Peters et al., 2007), but with $\mathbf{A}\ddot{\mathbf{q}} - \mathbf{b}$ imposed as soft-constraint and not as hard constraint. For $\Sigma_{\ddot{\mathbf{x}}} = 0$, we obtain a hard constraint and all the control laws in (Peters et al., 2007) can be recovered. The optimization view does not provide a direct way to update the joint covariance $\Sigma_{\ddot{\mathbf{q}}}$ if several tasks need to be prioritized, in contrast to the Bayesian approach.

4.4.1 Comparison to Strict Prioritization Approaches.

Our control law can be formulated for torques \mathbf{u} instead of desired accelerations $\ddot{\mathbf{q}}$. These derivations are given in the appendix. The mean $\boldsymbol{\mu}_{\mathbf{u}}$ of the controls, given in Equation (4.14), has a similar structure as well-known operational-space control laws (Peters et al., 2007; Khatib, 1987; Khatib et al., 2004; Nakamura et al., 1987; Sentis and Khatib, 2005; Park et al., 2002; Bruyninckx and Khatib, 2000; Luh et al., 1980). It consists of a model-based component to compensate for the dynamics of the system, the desired acceleration in the operational-space—which, for example, can be the output of a feedback controller—and a projection component $(\mathbf{I} - \mathbf{J}^\dagger \mathbf{J})$.

The difference to the aforementioned approaches lays in the computation of the generalized inverse matrix of the Jacobian \mathbf{J}^\dagger . By applying a Bayesian approach, we obtain a generalized inverse matrix of the Jacobian which is both regularized and weighted, while strict prioritization methods use an un-regularized inverse.

All these related approaches can be derived by assuming that the operational-space variance $\Sigma_{\ddot{\mathbf{x}}}$ is zero, i.e. $\Sigma_{\ddot{\mathbf{x}}} = \lim_{\alpha \rightarrow 0} \alpha \mathbf{I}$ and, therefore, degrade our approach to a deterministic case. If the operational-space variance is zero, the matrix $\mathbf{J}\mathbf{J}^\dagger = \mathbf{I}$ of the projection is a null-space projection, i.e. the lower priority tasks will not interfere with the higher priority tasks. Therefore, decreasing the variance of the operational-space controller $\Sigma_{\ddot{\mathbf{x}}}$ can be interpreted as “hardening” the prioritization of the two controllers.

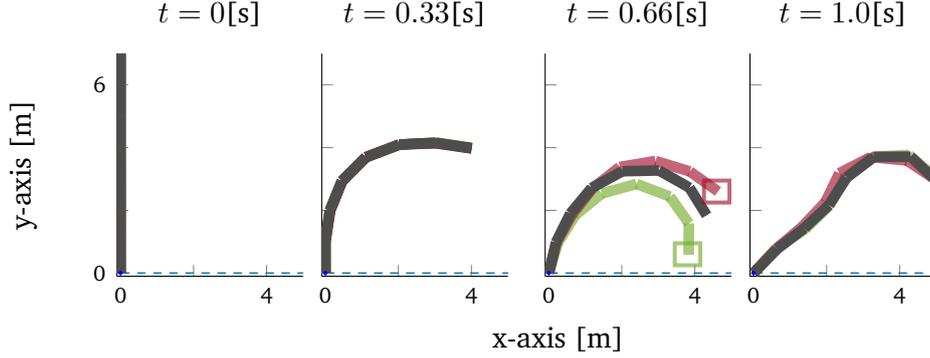


Figure 4.1: A visualization of the 7-link planar robot trajectory for different time-steps. The dark configuration denotes the mean reproduction of the demonstrated primitive. In green and red we show the mean reproduction after conditioning at $t = 0.66s$ to different via-points in end-effector space. The boxes show the targets of the end-effector.

A consequence of not regularizing the generalized inverse is the numerical instability of the inversion at singular kinematic configurations. A regularization of the form $\lambda \mathbf{I}$ is commonly used (Baerlocher and Boulic, 1998). In our approach, this regularization has the physical interpretation of adapting the covariance of the operational space task. To our knowledge, the interpretation of this regularization has not been previously discussed.

By setting the joint-space covariance to $\Sigma_{\ddot{q}} = \mathbf{I}$, the pseudo inverse is unregularized and unweighted and we can obtain controls laws as in (Peters et al., 2007; Khatib, 1987; Khatib et al., 2004; Nakamura et al., 1987; Sentis and Khatib, 2005; Park et al., 2002; Luh et al., 1980). Setting the joint-space covariance to $\Sigma_{\ddot{q}} = \mathbf{M}^{-1}$, we obtain controllers based on the Gauss principle of least constraint, and consistent to d’Alambert’s principle of virtual work (Bruyninckx and Khatib, 2000; Peters et al., 2007). The different approaches for computing the generalized inverse are shown in Table 4.1.

4.5 Experimental Evaluation

We evaluate our approach on redundant simulated and physical robots combining tasks learned by demonstrations. First, we demonstrate that our approach can be used for conditioning in operational space, an operation that was not feasible for the original ProMP approach. Second, we show that additional controllers can be smoothly integrated in our framework to implement additional constraints that were not present during training, e.g., for avoiding obstacles or keeping contact with another object. Third, we show how we learn a combination of tasks with considerably improved data efficiency and even generalize to unseen combinations.

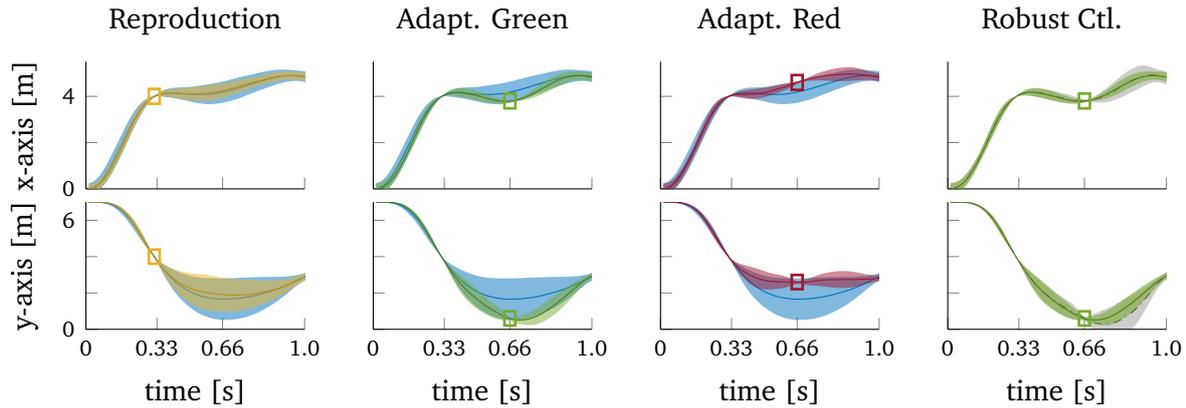


Figure 4.2: We present the end-effector trajectory distribution of the 7-link planar robot evaluation. The demonstrated distribution is shown in blue. The shaded area represents two times the standard deviation. In the first column, the reproduced trajectory distribution, shown in yellow, follows accurately the demonstrations. The boxes illustrate the via-points present in the demonstrations. In the second and third columns, we present the reproduction distribution after adapting the primitive. The boxes illustrate the conditioning points. In the fourth column, we evaluate the performance of the proposed feedback controller. The reproduction distribution is shown in green and the reproduction of the original controller is shown in gray.

4.5.1 Data-Driven Task-Space Adaptation

In this section, we adapt the learned task-space movement without explicitly solving the inverse kinematics problem. The adaptation, instead, is data-driven with the resulting trajectories staying close to the demonstrations. For the evaluation, we used a planar robot with seven degrees of freedom (DoFs) and optimal control to generate demonstrations in joint-space. Optimal control allows for generating trajectories under the assumption that the cost function of the task is known. For a real application though, the cost function is typically not available.

We used ten demonstrations for training a joint-space ProMP and a task-space ProMP. The movement of the robot is visualized in Fig. 4.1. The demonstrations have different variability at different time-steps of the movement, e.g., at time step $t = 0.33s$ the end-effector has low variability in both task-space dimensions. First, in Fig. 4.2 we show that our proposed controller can accurately track the demonstrated movement. Further, we adapt the task-space primitive by conditioning. Due to our prioritization scheme, no inverse kinematics algorithm is needed to find the respective joint configuration. Instead, the inverse kinematics problem is solved implicitly, in a data-driven way, using the prioritized controllers. We present our results in Fig. 4.1 and 4.2, where we adapt the learned primitive for two via-points. The reproduction can accurately pass through the via-points while it maintains the shape of the movement learned from the demonstrations. We compare the performance of the proposed

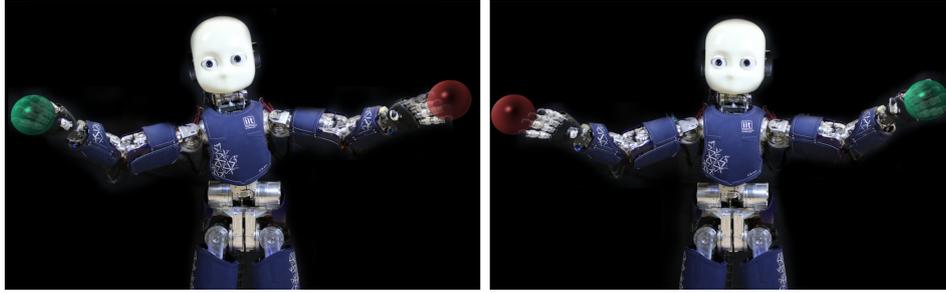


Figure 4.3: The iCub robot performing a bi-manual reaching task while its “hip” stays fixed. The importance of the targets is time-varying, allowing the robot to perform all tasks.

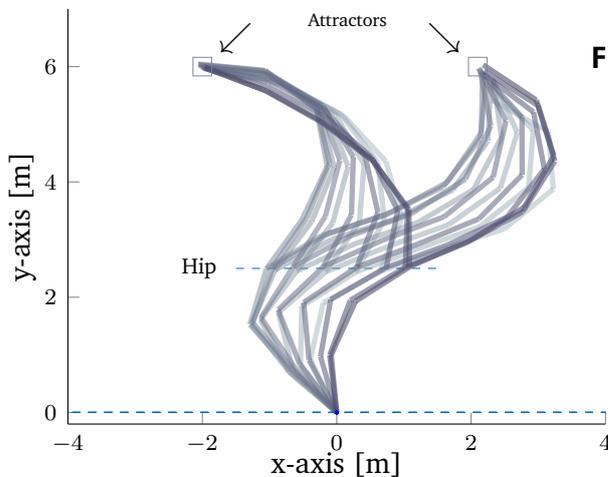


Figure 4.4: The planar robot performing a bi-manual reaching task while moving its “hip”, by combining three tasks, the two end-effector tasks and the hip task. The robot accurately stays at the desired targets with its end-effectors during the movement while the “hip” tracks a trajectory to remain at constant height.

robust feedback controller to the controller proposed in (Paraschos et al., 2013a) in Fig. 4.2, where the latter shows an improved tracking performance.

4.5.2 Incorporation of External Control Laws

Expert-knowledge can be incorporated in our approach to adapt the learned primitives. We demonstrate our approach on a planar robot with two end-effectors that has three links for the torso and five links for each arm. Each link is one meter long. The robot tracks a “hip” movement that stays at a constant height of $2.5m$. An expert designed two feedback controllers with high gains and low variance $\Sigma_u = 10^3$ that attract end-effectors at $\{2, 6\}(m)$ and $\{-2, 6\}(m)$, respectively. The resulting movement is shown in Fig. 4.4, where the robot performs successfully all of the three tasks; it reproduces accurately the hip movement staying at the desired height and places its end-effectors at the desired locations set by the expert.

Similarly, we evaluated our approach on the “iCub” robot. We defined a target for each hand of the robot to be reached with high accuracy at different time points. The hip of the robot should remain fixed for maintaining balance. The robot can successfully perform all tasks, as shown in Fig. 4.3.

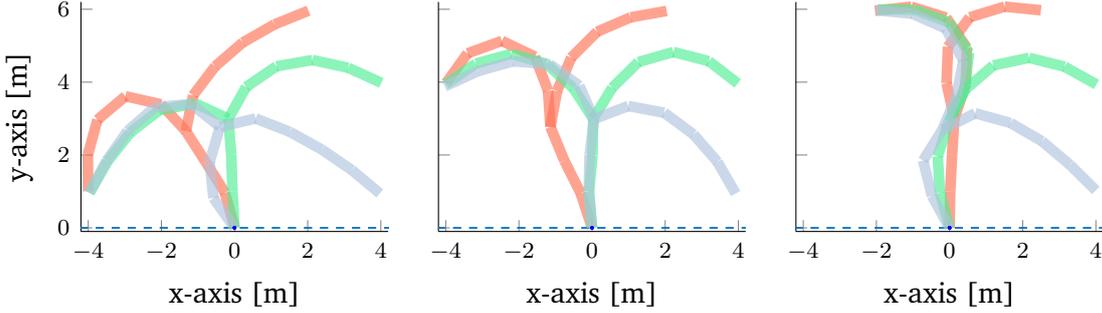


Figure 4.5: A visualization of the two end-effector robot. We present the final configuration, $t = t_{\text{end}}$, of the robot in task space for all nine different tasks.

4.5.3 Increased data efficiency

We demonstrate the increased learning efficiency of our approach in combining tasks of different end-effectors, over the traditional approach of learning all task combinations independently. For our experiments we used the planar robot with two end-effectors and thirteen DoF. The two concurrent end-effector tasks are not independent, as they control the common joints of the torso.

First, we generated demonstrations where each end-effector reaches one out of three end-points at $t_{\text{end}} = 1$. The end-point can either be set at $\{\pm 4, 1\}$, $\{\pm 4, 4\}$, or $\{\pm 2, 6\}$. An illustration of the configuration of the robot at these points is shown in Fig. 4.5. The combination of all three tasks of the two end-effectors yields nine different task combinations. For each combination, we generate a set of noisy demonstrations. As a baseline, we train nine individual primitives, one for each combination of tasks. In contrast, our approach can use all available demonstrations per task of a single end-effector as it can learn the end-effector tasks independently resulting in three times as much training data per task. Therefore, our approach considerably outperforms the baseline as the distributions can be estimated with higher accuracy, as shown in Fig. 4.6. We evaluate the average performance of both approaches which was specified as the negative square deviation from the true desired task-space position at the end of the movement.

4.5.4 Initiating Contacts during Reaching

We performed a bi-manual experiment using the “iCub” to reach objects while improving its stability by partially supporting its weight on a table. The experiment is difficult to demonstrate using kinesthetic teaching, as the teacher would have to simultaneously move both arms of the robot and keep track of the torso configuration. However, using the decoupling properties of our approach the task can be demonstrated with ease, a single arm at a time. Additionally, the decoupling approach utilizes improved data-efficiency, as it was shown in Sec. 4.5.3, and allows to generalize in novel movement combinations.

Reaching objects that require the robot to bend the torso can move the center of gravity of the robot out of the support polygon defined by the feet, and, as a result, the

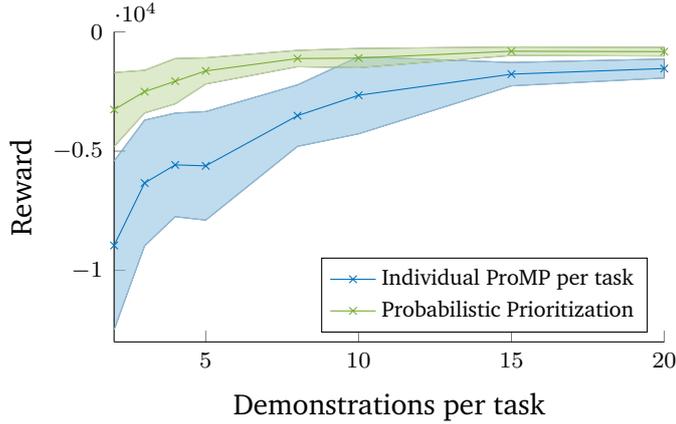


Figure 4.6: Comparison between the prioritization of MPs and learning each task combination independently, used as a baseline. We vary the number of demonstrations used per task combination. Using prioritization, we can learn the tasks for each end-effector independently, and, therefore, use more training data per task. The baseline can only learn each combination of the end-effector tasks individually. The plot shows the average negative square deviation from the true end-effector position averaged over ten trials.

Table 4.2: Initiating Contacts during Reaching Evaluation

	Left Task Err. (cm)	Right Task Err. (cm)
Blue — Marker	2.38 ± 0.91	3.09 ± 1.22
Blue — Ball	2.34 ± 0.96	3.18 ± 1.10
Blue — Cake	2.05 ± 0.71	3.56 ± 1.45
Green — Marker	2.21 ± 0.64	1.70 ± 0.81
Green — Ball	2.47 ± 0.89	2.28 ± 1.26
Green — Cake	2.97 ± 0.84	3.85 ± 1.02
Red — Marker	3.67 ± 0.76	2.89 ± 1.66
Red — Ball	2.82 ± 0.75	2.43 ± 1.13
Red — Cake	3.31 ± 1.26	4.23 ± 1.62

robot will lose its balance. The task of the robot is to perform a reaching movement while it initiates a contact to stabilize itself. With its right arm reaches for three different objects, as shown in Fig. 4.7. Concurrently, with the left arm, initiates a contact with the table that increases its stability. The location of the contact varies over three positions. We provided ten demonstrations for reaching each object and for each supportive contact location. The robot was capable of reproducing the movements using the prioritized movement primitives and generalizing to task combinations not present in the demonstrations. The reaching performance is shown in Table 4.2.

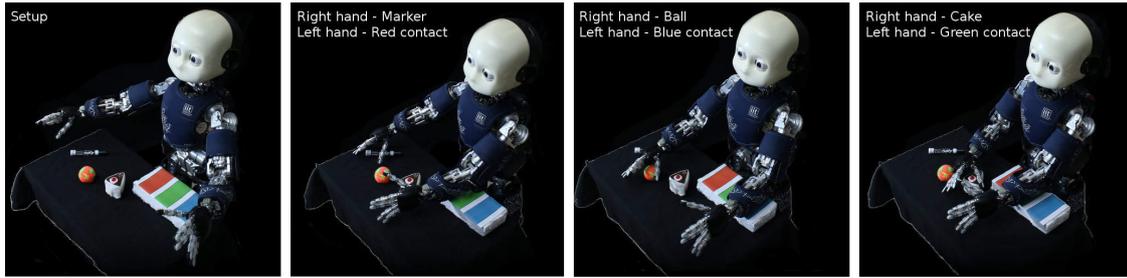


Figure 4.7: The iCub robot performing a bi-manual reaching task. With the left end-effector, the robot initiates a supportive contact with the environment, while it performs a reaching task with the right end-effector. We illustrated our setup in the first picture. The robot stands on the floor and can choose three supporting contact locations, shown in blue, green, and red. With the right end-effector the robot reaches for one of three different objects, a marker, a ball, or a piece of cake. In the remaining pictures we present our results for learning and generalizing to new task combinations of reaching and contact support locations.

4.5.5 Adaptation of a Movement Primitive Library

We demonstrate how to adapt a movement primitive library to changes of the environment, e.g., an obstacle that was not present during training. In our setup, Fig. 4.8, we used a seven DoF humanoid arm mounted on a fixed base, with a hand as its end-effector. The hand was not controlled during the experiments. We trained the robot with three distinct movements using kinesthetic teaching. First, approach a bottle for grasping, second, drop a peg into a bowl, and, third, push a button. Each primitive can generalize into a $25\text{cm} \times 25\text{cm}$ area. We provided ten demonstrations for each primitive. After training the primitives, we introduce an obstacle, the cat, into the environment. The robot collides with the cat during the execution of the primitives. To adapt the library, we demonstrated an additional primitive that avoids collisions with the obstacle, but without any of the three objects present in the environment during the demonstrations. While the robot does not perceive the obstacle, the information of avoiding it is encoded in the new primitive. By combining the new primitive with each of the primitives in the library, we avoid collisions with the obstacle, as shown in Fig. 4.9. The success rate of the experiment is presented in Table 4.3, averaging over ten reproductions per case. Finally, we evaluated classical prioritization approaches on our system. We modelled the mean trajectory using ProMPs, set $\Sigma_{\dot{q}} = \mathbf{I}$, and $\Sigma_{\ddot{x}} = \lambda \mathbf{I}$. The value of λ was optimized to reduce the Cartesian error at the end of the movement. The tasks could not be effectively combined using standard methods (Peters et al., 2007; Khatib, 1987; Khatib et al., 2004; Nakamura et al., 1987; Sentis and Khatib, 2005; Park et al., 2002; Bruyninckx and Khatib, 2000; Luh et al., 1980; Baerlocher and Boulic, 1998), as shown in Table 4.3.

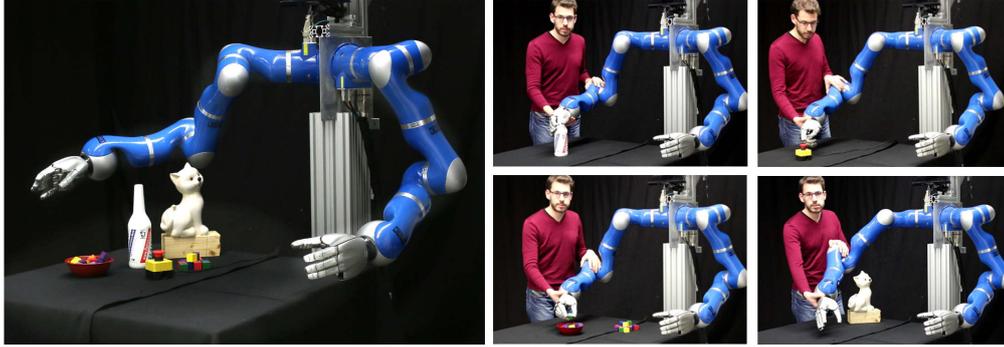


Figure 4.8: The setup used for adapting a primitive library due to environmental changes. The robot learns how to grasp a bottle, place a peg into a bowl, and push a button with kinematic teaching. The cat is then placed in a position that collides with the robot during reproduction.

Table 4.3: Adaptation of a MP Library — Task Evaluation

	Bottle	Peg	Button
Obstacle Avoidance Rate			
Before Adaptation	0.0	0.1	0.3
After Adaptation	0.9	1.0	1.0
Std. Approaches	1.0	1.0	1.0
Avg. End-Effector Error (cm)			
After Adaptation	0.48 ± 0.23	0.67 ± 0.23	0.45 ± 0.51
Std. Approaches	12.83 ± 4.57	3.21 ± 1.45	2.12 ± 1.53

4.6 Epilogue

In this chapter, we presented a novel approach for movement prioritization based on the combination of Bayesian task prioritization and the Probabilistic Movement Primitives. While prioritization is a well established concept in control, it has not been explored in the context of learning movements from demonstrations. We have shown that combining prioritization with learning approaches yields in a powerful representation that can be used to solve a combination of tasks with different end-effectors. Our approach is data-driven, i.e., it can solely be trained from demonstrations and minimizes expert knowledge. It avoids the problem of specifying a cost function for the task at hand, which is still an open problem. We demonstrated that our approach can be used to adapt task-space movements without solving an inverse kinematics problem and, importantly, staying close to the demonstrated data. Further, we propose an extension to the ProMP controller that can handle deviations that occur from the demonstrated movements due to the prioritization.

A key contribution of our approach is the ability to combine tasks of different end-effectors in a principled and data-efficient way. Our approach can generalize to task combinations that were not present in the demonstrations and requires significantly

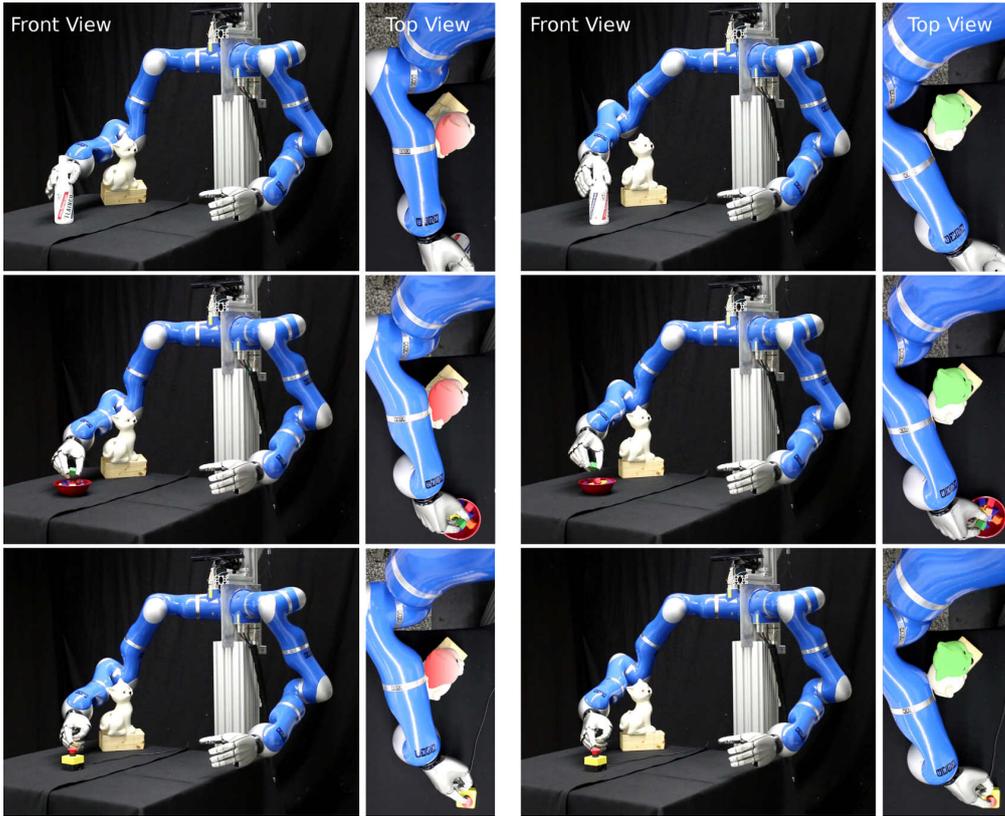


Figure 4.9: In the left, each row shows the reproduction of the respective primitive after training and the introduction of the obstacle without adaptation. The robot collides with the head of the cat. In the right, we present the reproduction after adaptation where the robot avoids collisions.

less training data to achieve the same level of performance. Our approach can be used to adapt a library of primitives without extensive retraining.

Given that our approach is data-driven, it heavily relies on quality demonstrations. If the task is too difficult to be demonstrated or if non-informative demonstrations are provided, our approach will match the provided data and not the intention of the user.

In future work, we will expand the evaluations of our approach on more complex real-world scenarios. We consider multiple task execution with physical robot interaction under the presence of contacts as interesting research direction.

4.A Including the Dynamics of the System

The stochastic controller on the joint acceleration given in Equation (4.6) can be used to control a physical system, i.e., by torque control, using the rigid-body dynamics model (Featherstone, 2014), $\mathbf{u} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q})$, where $\mathbf{M}(\mathbf{q})$ denotes the inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ denotes Coriolis and centripetal forces, and $\mathbf{G}(\mathbf{q})$ forces due to gravity. Using the rigid-body dynamics model, we reformulate our controller

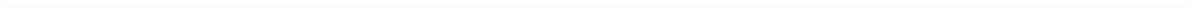
to operate in the joint torque space, i.e. $p_{1|2}(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\boldsymbol{\mu}'_u, \boldsymbol{\Sigma}_u)$. The mean $\boldsymbol{\mu}_u$ of this controller is given by

$$\boldsymbol{\mu}'_u = \mathbf{M} \left(\mathbf{J}^\dagger \left(\boldsymbol{\mu}_{\ddot{\mathbf{x}}} - \dot{\mathbf{J}}\dot{\mathbf{q}} \right) + (\mathbf{I} - \mathbf{J}^\dagger \mathbf{J}) \boldsymbol{\mu}_{\ddot{\mathbf{q}}} \right) + \mathbf{C} + \mathbf{G},$$

where we used $\boldsymbol{\mu}_{\ddot{\mathbf{q}}} = \mathbf{M}^{-1}(\boldsymbol{\mu}_u - \mathbf{C} - \mathbf{G})$. Furthermore, decoupling of the kinematics and the dynamics can be obtained by setting $\hat{\boldsymbol{\mu}}_u = \boldsymbol{\mu}_u + \mathbf{C} + \mathbf{G}$ and using it in place of $\boldsymbol{\mu}_u$. In this case, the mean becomes

$$\boldsymbol{\mu}'_u = \mathbf{M} \mathbf{J}^\dagger (\ddot{\mathbf{x}} - \dot{\mathbf{J}}\dot{\mathbf{q}}) + \mathbf{M}(\mathbf{I} - \mathbf{J}^\dagger \mathbf{J})(\mathbf{M}^{-1} \hat{\boldsymbol{\mu}}_u) + \mathbf{C} + \mathbf{G} \quad (4.14)$$

that results in the resolved-acceleration controller (Yoshikawa, 1990; Hsu et al., 1988).



5 Conclusion and Future Work

5.1 Summary of Contributions

The research described in this thesis aims at developing a generic framework for learning, representing, and executing complex movement skills in redundant robots. We have considered several important challenges to the generalization of the movement concerning the reusing of learned skills to new situations, combination of skills to solve more complex problems, and skills where the robot physically interacts with the environment. We train probabilistic models from human demonstrations as a mean of transferring expert knowledge from humans to the robot. Furthermore, by modelling the skill’s uncertainty, our approach automatically identifies significant parts of the skill from data, e.g. via-points, and, therefore, minimizes manual data processing and fine-tuning.

Specifically, in Chapter 2, we focused on representing the skill, rapidly training the robot to acquire new skills, and using it in real world scenarios. We have developed the basis of our framework, the Probabilistic Movement Primitives (ProMPs), a concise movement representation that encodes the skill’s uncertainty. ProMPs can be trained from human demonstrations. Operating in unstructured environments, robots should be capable to adapt learned skills to new situations and reproduce them in a safe manner. We have derived novel adaptation operations where, using the skill’s uncertainty, we adapt the skill ensuring that they stay “close” to the demonstrations. To control the physical system, we have derived analytically a stochastic feedback controller with time-varying gains. The gains are being modulated by the skill’s uncertainty at every time point.

In Chapter 3, we investigated the problem of reproducing skills that involve physical interaction with objects. The object dynamics might differ from the dynamics the robot previously experienced during training. For example, moving a bottle of water filled to a different level. We have developed a model-free approach of the Probabilistic Movement Primitives (ProMP) that learns skills physically interacting with the environment, from human demonstrations. After the teacher demonstrates how to perform the skill with a few examples, the robot is able to generalize to new scenarios encountered during reproduction. Our approach neither requires a known model of the system dynamics nor attempts to explicitly train one. Rather, we correlate the actions presented during the training to the state of the robot. Similar to the model-based approach, we have also derived a stochastic feedback controller with time-varying gains in closed-form for controlling the physical system. The proposed controller generates actions for the encountered dynamics and generalizes well within the vicinity of the demonstrations. We demonstrated that force/torque sensing is also necessary for the robot to accurately perform physical interaction tasks. Dur-

ing the reproduction of a skill, the robot exerts interaction forces similar to the ones demonstrated.

In the last part of the thesis, Chapter 4, we focused on solving more complex tasks by simultaneously activating multiple primitives. Standard approaches (Peters et al., 2007; Khatib, 1987; Khatib et al., 2004; Nakamura et al., 1987; Sentis and Khatib, 2005; Park et al., 2002; Bruyninckx and Khatib, 2000; Luh et al., 1980; Baerlocher and Boulic, 1998) heavily rely on expert knowledge and require manual tuning for setting the activation of each primitive. Alternatively, iterative learning methods (Lober et al., 2014, 2016), e.g., reinforcement learning, have been used to optimize the activation parameters. In our proposed approach, the activation of each primitive is directly learned from demonstrations and does not require any additional manual tuning. The uncertainty of each skill is then used to modulate the combination. In contrast to standard approaches, when the skill’s uncertainty is high, the robot is allowed to deviate from the mean trajectory in order to execute other skills. Therefore, our approach allows to perform skill combinations that were not possible with standard approaches. Further, solving complex tasks with the combination of skills inherently distributes the task’s complexity. Simpler skills can be learned independently with fewer demonstrations, improving data efficiency. Further, learning and combining independent skills results in a modular control architecture, where skills can be enabled or disabled on demand.

5.2 Discussion and Future Work

During this thesis, we have developed a skill learning framework that allows for accurate reproduction of skills in real world environments. In this section, we critically review open problems and promising directions for future research.

Optimization of the open parameters

The proposed movement primitives framework simplifies training by encoding full motion sequences of various skills directly from demonstrations without requiring experts to segment the demonstrated trajectories. However, our approach still depends on open parameters that have to be tuned by hand. Specifically, during our experimentation, we adjusted parameters associated with the basis functions, i.e., the type, the number of basis, the centers, and the bandwidth. In some settings, additional parameters for the stabilizing control were manually chosen. As the ProMPs are a generative model, the parameters associated with the basis functions can be optimized off-line. This is done by defining a cost function to minimize the distance between the observed trajectory distribution of the demonstrations and the generated one, e.g. by minimizing the KL divergence. An additional optimization objective that reduces the complexity of the model could be used to avoid over-fitting, for example using a L1 norm on the parameters, and facilitate reinforcement learning approaches by learning a low dimensional representation. The parameters of the additional stabilizing control laws, though, can not be optimized off-line as they require roll-outs in a realistic simulation or a physical system. The training in this case should strive to reach a stable behavior, while the activation of the additional controllers is minimized.

Non-parametric representation

An alternative approach to optimizing the open parameters of our framework is to incorporate non-parametric approaches. Specifically, the linear basis function model can be substituted with a non-parametric model. The non-parametric approaches can fit the demonstrated data by increasing the resolution of the approach to the points of interest. The increased flexibility of non-parametric approaches comes with an increased computational cost and an additional optimization of hyper-parameters. The transition to a non-parametric model will require new derivations of the stochastic feedback controller. Currently, the controller is derived under the assumption that the trajectory distribution per time step can be modelled with a Gaussian distribution. A straight-forward approach would be to fit a Gaussian distribution to the output of the non-parametric model.

Using Mixture of Models for Control

The Gaussian assumption for skills might not always be accurate, especially in cases where skills have multiple solutions. In the latter case, the presented framework would average the solutions and might fail to successfully reproduce the skill. An extension using a mixture of Gaussian has been proposed in (Rueckert et al., 2015a; Ewerton et al., 2015b). The authors replaced the Gaussian distribution that describes the weights of the basis function model with a mixture of Gaussian. However, the authors did not consider controlling the physical system and no control laws were derived. An interesting future direction would be to extend the stochastic feedback controller by employing mixture models, to ensure the continuity of the control signal when the active component of the mixture model changes.

Integrating Informative Priors

In this thesis, we have also assumed that a new skill can be successfully acquired from human demonstrations. However, knowledge previously acquired skills could be integrated in form of a prior to reduce the amount of demonstrations needed. Currently, our approach uses non-informative priors over the parameters for the skill. The prior could be selected by an expert, or generic priors could be developed. Task specific priors could be build by analyzing demonstrations from similar tasks. Generic priors could have properties that reflect general traits of human motion. An example of a generic prior could be to decorrelate time points that lay outside a time window, as, currently, our approach learns the temporal correlations for the whole duration of the movement. Such a prior, could facilitate learning by enabling a more accurate estimation of the covariance of the weights Σ_w .

Evaluating the Coherence of the Demonstrations

During teaching complex skills in our experiments, the natural case was the teacher demonstrating a non-optimal solution of the task to the robot. It would be beneficial

to automatically discover demonstrations that do not fit our model and, if used, deteriorate the performance of the proposed approach. Since we model the skills with Gaussian distributions, normality metrics of the acquired dataset can indicate problems with the dataset. A demonstration that does not fit a specific dataset can indicate that a separate primitive should be learned. This approach could be combined with clustering the demonstrations to optimize the number of clusters and adding context variables. All learned primitives could then be aggregated in primitive libraries and combined using our proposed approach.

Online Learning of Dynamical Models

In this thesis, we assumed that either the dynamics model of the robot is known or that the actions during the demonstrations can be observed. However, for many robots such a dynamic model may be hard to obtain due to the change of the robot dynamics over time, especially for cable driven robots where cables tend to elongate with use. Therefore, a method for learning the dynamics of the robot is needed. To better integrate such a method in our approach, the method should be able to provide a linearization of the system dynamics given the state of the robot, e.g. Bayesian locally weighted regression, or Gaussian processes. For complex systems, learning dynamic models can be time consuming. We propose to accurately model the dynamics only in the vicinity of the demonstrated movements. To accommodate the changing dynamics over time, the learning approach can operate online. Finally, if the learned model can provide a measure of confidence, we can use this uncertainty to modulate the proposed model-based controller. It can be shown that higher uncertainty in the system matrix will reduce the feedback gains generated by our controller.

Improving the Control Policy with Reinforcement Learning

As our approach is data-driven, the performance heavily depends on the quality of the demonstrations. In complex skills, the teacher can fail to demonstrate the skill successfully to the robot and could require multiple trials before it can successfully demonstrate the task. However, from our experience, the first attempts of the demonstrator could be used as an initial solution that support the human in solving the task. Reinforcement learning methods can be applied for adapting the demonstrated trajectory distribution to successfully reproduce the skill. However, these approaches assume that the reward function of that task is known, which, in general, is not the case.

Perceptual coupling

Movement coordination of external and often not directly controllable objects is essential for successfully performing many tasks. For example, catching a thrown ball or bi-manually lifting a box requires coordinated actions (Calinon et al., 2013; Rozo et al., 2013; Gams et al., 2014). While our approach can encode the correlations between the degrees of freedom of the robot, or force/torque sensing, we have not

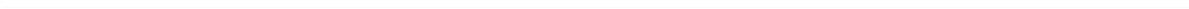
considered perceptual coupling scenarios. Apart from correctly learning the correlations of such tasks, perceptual coupling poses the challenging aspect of inferring the phase accurately as timing is critical for these tasks (Kober et al., 2008). Additionally, to perform control all operations must be completed within hard real-time constraints which is restricting the estimation approaches, such as the one presented in Chapter 3.

Imitation Learning from Camera Images

Imitation learning in general is not restricted to kinesthetic teaching and teleoperation. Camera images have been used as an input source for teaching the robot (Gams et al., 2010). Camera images or related RGBD systems can be used in our approach for learning an initial solution of the task. For example, skeleton tracking can be extracted from those. Alternatively, the camera image could be fed into a deep neural network to learn features, a low dimensional representation of the task, and ProMPs can be trained in the low dimensional space. However, reinforcement learning in this case would be required in order to find a policy that reproduces the learned skill.

5.3 Outlook

The skill learning framework presented in this thesis focused on enabling robots to be more autonomous by proving approaches for rapid novel skill acquisition, skill adaptation, and reproduction. The proposed future work aims on exploring new application domains for the skill acquisition framework and using a variety of modalities for learning new skills. Additionally, the future work aims on automating the framework to consistently achieve high performance without manual tuning of the open parameters. Together with the autonomous optimization of the learned skills, it could potentially let robots to better integrate into our every day lives.



Publication List

Excerpts of the research presented in this thesis have led to the following publications.

Articles under Review

Paraschos, A., Daniel, C., Peters, J., and Neumann, G. Using probabilistic movement primitives in robotics. *submitted to Autonomous Robots* (2015).

Paraschos, A., Lioutikov, R., Peters, J., and Neumann, G. Imitation learning under unknown system dynamics. *submitted* (2017a).

Paraschos, A., Lioutikov, R., Peters, J., and Neumann, G. Probabilistic prioritization of primitives. *submitted to Robotics and Automation Letters* (2017b).

Journal Articles

Englert, P., Paraschos, A., Peters, J., and Deisenroth, M. P. Probabilistic model-based imitation learning. *Adaptive Behavior Journal*, pages 388–403 (2013). [[pdf](#)].

Lioutikov, R., Paraschos, A., Neumann, G., and Peters, J. Generalizing movements with information theoretic stochastic optimal control. *Journal of Aerospace Information Systems* (2014).

Neumann, G., Daniel, C., Paraschos, A., Kupcsik, A., and Peters, J. Learning modular policies for robotics. *Frontiers in Computational Neuroscience* (2014).

Refereed Conference Articles

Englert, P., Paraschos, A., Peters, J., and Deisenroth, M. P. Model-based imitation learning by probabilistic trajectory matching. In *Proceedings of 2013 IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany* (2013). [[pdf](#)].

Lioutikov, R., Paraschos, A., Neumann, G., and Peters, J. Sample-based information-theoretic stochastic optimal control. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China* (2014). [[pdf](#)].

-
- Paraschos, A., Daniel, C., Peters, J., and Neumann, G. Probabilistic movement primitives. In *Advances in Neural Information Processing Systems (NIPS)*, Cambridge, MA, Lake Tahoe, USA (2013a). [pdf].
- Paraschos, A., Neumann, G., and Peters, J. A probabilistic approach to robot trajectory generation. In *Proceedings of the International Conference on Humanoid Robots (HUMANOIDS)*, Atlanta, USA (2013b). [pdf].
- Paraschos, A., Rueckert, E., Peters, J., and Neumann, G. Model-free probabilistic movement primitives for physical interaction. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany (2015). [pdf].
- Parisi, S., Abdulsamad, H., Paraschos, A., Daniel, C., and Peters, J. Reinforcement learning vs human programming in tetherball robot games. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany (2015). [pdf].
- Rueckert, E., Mundo, J., Paraschos, A., Peters, J., and Neumann, G. Extracting low-dimensional control variables for movement primitives. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, Seattle, USA (2015). [pdf].
- Tanneberg, D., Paraschos, A., Peters, J., and Rueckert, E. Deep spiking networks for model-based planning in humanoids. In *Proceedings of the International Conference on Humanoid Robots (HUMANOIDS)*, Cancun, Mexico (2016). [pdf].

Other Published Articles

- Deisenroth, M. P., Englert, P., Paraschos, A., and Peters, J. Imitation learning by model-based probabilistic trajectory matching. In *Proceedings of Machine Learning and Cognitive Science (MLCogSys)*, Palma de Mallorca, Spain (2013).
- Deisenroth, M. P., Englert, P., Paraschos, A., Peters, J., Rasmussen, C. E., and Fox, D. Autonomous planning and control with bayesian nonparametric models. In *NIPS Workshop on Bayesian Nonparametric Models (BNPM) For Reliable Planning And Decision-Making Under Uncertainty*, Lake Tahoe, USA (2012).
- Englert, P., Paraschos, A., Peters, J., and Deisenroth, M. P. Addressing the correspondence problem by model-based imitation learning. In *Autonomous Learning Workshop, IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany (2013).
- Neumann, G., Paraschos, A., and Peters, J. Probabilistic movement primitives. In *2nd NSF Workshop on Formal Composition of Motion Primitives*, Philadelphia, USA (2013).
- Paraschos, A., Neumann, G., and Peters, J. Optimal movement primitives. In *Machine Learning Summer School, La Palma, Canary Islands, Spain* (2012).

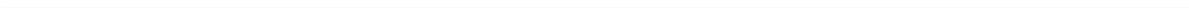
Paraschos, A., Neumann, G., and Peters, J. Probabilistic Movement Primitives (ProMPs). In *Workshop on Novel Methods for Learning and Optimization of Control Policies and Trajectories for Robotics, IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany* (2013a).

Paraschos, A., Neumann, G., and Peters, J. Representing the variability of human movement with probabilistic movement primitives. In *Workshop on Benchmarking Reaching Motion Algorithms, International Conference on Humanoid Robots (HUMANOIDS), Atlanta, USA* (2013b).

Technical Reports

Abdulsamad, H., Buchholz, T., Croon, T., Khoury, M. E., and Paraschos, A. Playing tetherball with compliant robots. Tech. rep., Advanced Design Project, Intelligent Autonomous Systems, TU-Darmstadt (2014). [[pdf](#)].

Ho, D., Kisner, V., and Paraschos, A. Trajectory tracking controller for a 4-dof flexible joint robotic arm. Tech. rep., Advanced Design Project, Intelligent Autonomous Systems, TU-Darmstadt (2014). [[pdf](#)].



Bibliography

- Baerlocher, P. and Boulic, R. Task-priority formulations for the kinematic control of highly redundant articulated structures. In *Int. Conf. on Intelligent Robots and Systems (IROS)* (1998).
- Billard, A., Calinon, S., Dillmann, R., and Schaal, S. Robot programming by demonstration. In *Springer handbook of robotics*, pages 1371–1394. Springer (2008).
- Bruno, D., Calinon, S., Malekzadeh, M. S., and Caldwell, D. G. Learning the stiffness of a continuous soft manipulator from multiple demonstrations. In *Intelligent Robotics and Applications*, pages 185–195. Springer (2015).
- Bruyninckx, H. and Khatib, O. Gauss’ principle and the dynamics of redundant and constrained manipulators. In *International Conference on Robotics and Automation (ICRA)* (2000).
- Buchli, J., Stulp, F., Theodorou, E., and Schaal, S. Learning variable impedance control. *The International Journal of Robotics Research*, 30(7):820–833 (2011).
- Calinon, S. A tutorial on task-parameterized movement learning and retrieval. *Intelligent Service Robotics*, 9(1):1–29 (2016).
- Calinon, S. and Billard, A. Statistical learning by imitation of competing constraints in joint space and task space. *Adv. Robot.*, 23(15):2059–2076 (2009).
- Calinon, S., Bruno, D., and Caldwell, D. G. A task-parameterized probabilistic model with minimal intervention control. In *Int. Conf. on Robotics and Automation (ICRA)*, pages 3339–3344 (2014).
- Calinon, S., D’halluin, F., Sauser, E. L., Caldwell, D. G., and Billard, A. G. Learning and reproduction of gestures by imitation. *Robotics and Automation Magazine* (2010a).
- Calinon, S., Kormushev, P., and Caldwell, D. Compliant Skills Acquisition and Multi-Optima Policy Search with EM-based Reinforcement Learning. *Robotics and Autonomous Systems (RAS)*, 61(4):369 – 379 (2013).
- Calinon, S., Li, Z., Alizadeh, T., Tsagarakis, N. G., and Caldwell, D. G. Statistical dynamical systems for skills acquisition in humanoids. In *Int. Conf. on Humanoid Robots*, pages 323–329 (2012).
- Calinon, S., Sardellitti, I., and Caldwell, D. G. Learning-based control strategy for safe human-robot interaction exploiting task and robot redundancies. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 249–254 (2010b).
- da Silva, B., Konidaris, G., and Barto, A. Learning Parameterized Skills. In *International Conference on Machine Learning* (2012).

-
- Daniel, C., Neumann, G., and Peters, J. Learning Concurrent Motor Skills in Versatile Solution Spaces. In *IEEE/RSJ International Conference on Intelligent Robots and Systems* (2012).
- dAvella, A. and Bizzi, E. Shared and Specific Muscle Synergies in Natural Motor Behaviors. *Proceedings of the National Academy of Sciences (PNAS)*, 102(3):3076–3081 (2005).
- Decre, W., Smits, R., Bruyninckx, H., and Schutter, J. D. Extending iTaSC to support inequality constraints and non-instantaneous task specification. In *Int. Conf. on Robotics and Automation (ICRA)* (2009).
- Degallier, S., Righetti, L., Gay, S., and Ijspeert, A. Toward simple control for complex, autonomous robotic applications: combining discrete and rhythmic motor primitives. *Autonomous robots*, 31:155–181 (2011).
- Dominici, N., Ivanenko, Y. P., Cappellini, G., d’Avella, A., Mondì, V., Cicchese, M., Fabiano, A., Silei, T., Di Paolo, A., Giannini, C., et al. Locomotor primitives in newborn babies and their development. *Science*, 334(6058):997–999 (2011).
- Ernesti, J., Righetti, L., Do, M., Asfour, T., and Schaal, S. Encoding of periodic and their transient motions by a single dynamic movement primitive. In *12th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 57–64 (2012).
- Evrard, P., Gribovskaya, E., Calinon, S., Billard, A., and Kheddar, A. Teaching physical collaborative tasks: object-lifting case study with a humanoid. In *2009 9th IEEE-RAS International Conference on Humanoid Robots*, pages 399–404. IEEE (2009).
- Ewerton, M., Maeda, G., Peters, J., and Neumann, G. Learning motor skills from partially observed movements executed at different speeds. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 456–463. IEEE (2015a).
- Ewerton, M., Neumann, G., Lioutikov, R., Ben Amor, H., Peters, J., and Maeda, G. Learning multiple collaborative tasks with a mixture of interaction primitives. In *ICRA* (2015b).
- Featherstone, R. *Rigid body dynamics algorithms*. Springer (2014).
- Forte, D., Gams, A., Morimoto, J., and Ude, A. On-line motion synthesis and adaptation using a trajectory database. *Robotics and Autonomous Systems*, 60:1327–1339 (2012).
- Gams, A., Do, M., Ude, A., Asfour, T., and Dillmann, R. On-line periodic movement and force-profile learning for adaptation to new surfaces. In *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*, pages 560–565 (2010).
- Gams, A., Nemeč, B., Ijspeert, A. J., and Ude, A. Coupling movement primitives: Interaction with the environment and bimanual tasks. *Robotics, IEEE Transactions on*, 30(4):816–830 (2014).

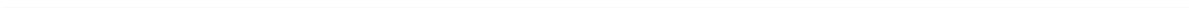
-
- Gribovskaya, E., Kheddar, A., and Billard, A. Motion learning and adaptive impedance for robot control during physical interaction with humans. In *2011 IEEE International Conference on Robotics and Automation*, pages 4326–4332. IEEE (2011).
- Higham, N. J. Computing a nearest symmetric positive semidefinite matrix. *Linear Algebra and its Applications*, 103 (1988).
- Hsu, P., Hauser, J., and Sastry, S. Dynamic control of redundant manipulators. In *Int. Conf. on Robotics and Automation* (1988).
- Ijspeert, A. J. Central pattern generators for locomotion control in animals and robots: a review. *Neural Networks*, 21(4):642–653 (2008).
- Ijspeert, A. J., Nakanishi, J., Hoffmann, H., Pastor, P., and Schaal, S. Dynamical movement primitives: learning attractor models for motor behaviors. *Neural Computation*, 25(2):328–373 (2013).
- Ijspeert, A. J., Nakanishi, J., and Schaal, S. Learning attractor landscapes for learning motor primitives. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1547–1554. MIT Press (2003).
- Kalakrishnan, M., Righetti, L., Pastor, P., and Schaal, S. Learning force control policies for compliant manipulation. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 4639–4644. IEEE (2011).
- Khansari-Zadeh, S. M. and Billard, A. Learning stable nonlinear dynamical systems with gaussian mixture models. *Transactions on Robotics* (2011).
- Khansari-Zadeh, S. M., Kronander, K., and Billard, A. Modeling robot discrete movements with state-varying stiffness and damping: A framework for integrated motion generation and impedance control. In *Robotics Science and Systems (R:SS)* (2014).
- Khatib, O. A unified approach for motion and force control of robot manipulators: The operational space formulation. *Journal of Robotics and Automation* (1987).
- Khatib, O., Sentis, L., Park, J., and Warren, J. Whole-body dynamic behavior and control of human-like robots. *International Journal of Humanoid Robotics* (2004).
- Klug, S., Lens, T., von Stryk, O., Möhl, B., and Karguth, A. Biologically inspired robot manipulator for new applications in automation engineering. In *Proceedings of Robotik* (2008).
- Kober, J., Mohler, B., and Peters, J. Learning perceptual coupling for motor primitives. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 834–839 (2008).
- Kober, J., Muelling, K., Kroemer, O., Lampert, C. H., Scholkopf, B., and Peters, J. Movement templates for learning of hitting and batting. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 853–858 (2010).
- Kober, J. and Peters, J. Learning motor primitives for robotics. In *Int. Conf. on Robotics and Automation (ICRA)* (2009).

-
- Kober, J. and Peters, J. Learning prioritized control of motor primitives. In *Learning Motor Skills*, pages 149–160. Springer (2014).
- Konidaris, G., Kuindersma, S., Grunert, R., and Barto, A. Robot Learning from Demonstration by Constructing Skill Trees. *International Journal of Robotics Research (IJRR)*, 31(3):360–375 (2012).
- Kormushev, P., Calinon, S., and Caldwell, D. G. Robot motor skill coordination with EM-based reinforcement learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3232–3237 (2010).
- Kormushev, P., Calinon, S., and Caldwell, D. G. Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input. *Adv. Robot.*, 25(5):581–603 (2011a).
- Kormushev, P., Nenchev, D. N., Calinon, S., and Caldwell, D. G. Upper-body kinesthetic teaching of a free-standing humanoid robot. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3970–3975 (2011b).
- Kronander, K. and Billard, A. Learning compliant manipulation through kinesthetic and tactile human-robot interaction. *IEEE Transactions on Haptics* (2013).
- Kulvicius, T., Ning, K., Tamosiunaite, M., and Worgotter, F. Joining movement sequences: Modified dynamic movement primitives for robotics applications exemplified on handwriting. *Robotics, IEEE Transactions on*, 28(1):145–157 (2012).
- Lazaric, A. and Ghavamzadeh, M. Bayesian Multi-Task Reinforcement Learning. In *Proceedings of the 27th International Conference on Machine Learning (ICML)* (2010).
- Lee, A. X., Lu, H., Gupta, A., Levine, S., and Abbeel, P. Learning force-based manipulation of deformable objects from multiple demonstrations. In *Int. Conf. on Robotics and Automation (ICRA)* (2015).
- Li, W. and Todorov, E. Iterative linear quadratic regulator design for nonlinear biological movement systems. In *In Proceedings of 1 st International Conference on Informatics in Control, Automation and Robotics* (2004).
- Lober, R., Padois, V., and Sigaud, O. Multiple task optimization using dynamical movement primitives for whole-body reactive control. In *Int. Conf. on Humanoid Robots (Humanoids)* (2014).
- Lober, R., Padois, V., and Sigaud, O. Variance modulated task prioritization in Whole-Body control. In *Int. Conf. on Intelligent Robots and Systems (IROS)* (2015).
- Lober, R., Padois, V., and Sigaud, O. Efficient reinforcement learning for humanoid whole-body control. In *International Conference on Humanoid Robots* (2016).
- Luh, J., Walker, M., and Paul, R. Resolved-acceleration control of mechanical manipulators. *Transactions on Automatic Control* (1980).

-
- Maeda, G., Ewerton, M., Lioutikov, R., Amor, H., Peters, J., and Neumann, G. Learning interaction for collaborative tasks with probabilistic movement primitives. In *Proceedings of the International Conference on Humanoid Robots (HUMANOIDS)*, pages 527–534 (2014).
- Matsubara, T., Hyon, S.-H., and Morimoto, J. Learning parametric dynamic movement primitives from multiple demonstrations. *Neural networks*, 24(5):493–500 (2011).
- Modugno, V., Neumann, G., Rueckert, E., Oriolo, G., Peters, J., and Ivaldi, S. Learning soft task priorities for control of redundant robots. In *Int. Conf. on Robotics and Automation (ICRA)* (2016).
- Moro, F. L., Tsagarakis, N. G., and Caldwell, D. G. On the kinematic motion primitives (kMPs) - theory and application. *Frontiers in Neurorobotics*, 6:10 (2012).
- Muelling, K., Kober, J., and Peters, J. Learning table tennis with a mixture of motor primitives. In *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*, pages 411–416 (2010).
- Muelling, K., Kober, J., and Peters, J. A biomimetic approach to robot table tennis. *Adaptive Behavior Journal*, (5) (2011).
- Mülling, K., Kober, J., Kroemer, O., and Peters, J. Learning to select and generalize striking movements in robot table tennis. *Int. J. Rob. Res.*, 32(3):263–279 (2013).
- Nakamura, Y., Hanafusa, H., and Yoshikawa, T. Task-Priority based redundancy control of robot manipulators. *International Journal of Robotics Research (IJRR)* (1987).
- Nakanishi, J., Morimoto, J., Endo, G., Cheng, G., Schaal, S., and Kawato, M. Learning from demonstration and adaptation of biped locomotion. *Robotics and Autonomous Systems*, 47:79–91 (2004).
- Neumann, G., Maass, W., and Peters, J. Learning Complex Motions by Sequencing Simpler Motion Templates. In *International Conference on Machine Learning (ICML)* (2009).
- O’Hagan, A. and Forster, J. Kendall’s advanced theory of statistics: Bayesian inference. 2b (2nd edn.), arnold, new york, ny. Tech. rep., ISBN 0-340-80752-0 (2004).
- Paraschos, A., Daniel, C., Peters, J., and Neumann, G. Probabilistic movement primitives. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Neural Inf. Proc. Sys.*, pages 2616–2624. Curran Associates, Inc. (2013a).
- Paraschos, A., Neumann, G., and Peters, J. A probabilistic approach to robot trajectory generation. In *“Int. Conf. on Humanoid Robots (Humanoids)* (2013b).
- Park, J., Chung, W.-K., and Youm, Y. Characterization of instability of dynamic control for kinematically redundant manipulators. In *International Conference on Robotics and Automation (ICRA)* (2002).

-
- Pastor, P., Hoffmann, H., Asfour, T., and Schaal, S. Learning and generalization of motor skills by learning from demonstration. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 763–768 (2009).
- Pastor, P., Righetti, L., Kalakrishnan, M., and Schaal, S. Online movement adaptation based on previous sensor experiences. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 365–371 (2011).
- Peters, J., Mistry, M., Udwadia, F., Nakanishi, J., and Schaal, S. A unifying framework for robot control with redundant DOFs. *Autonomous Robots* (2007).
- Peters, J., Mistry, M., Udwadia, F. E., Nakanishi, J., and Schaal, S. A Unifying Methodology for Robot Control with Redundant DOFs. *Autonomous Robots*, (1):1–12 (2008).
- Righetti, L. and Ijspeert, A. J. Programmable central pattern generators: an application to biped locomotion control. In *Proceedings of IEEE International Conference on Robotics and Automation, (ICRA)*, pages 1585–1590 (2006).
- Rozo, L., Calinon, S., Caldwell, D., Jiménez, P., and Torras, C. Learning collaborative impedance-based robot behaviors. In *AAAI Conference on Artificial Intelligence* (2013).
- Rückert, E. A., Neumann, G., Toussaint, M., and Maass, W. Learned graphical models for probabilistic planning provide a new class of movement primitives. *Frontiers in Computational Neuroscience*, 6:97 (2012).
- Rueckert, E., Mundo, J., Paraschos, A., Peters, J., and Neumann, G. Extracting low-dimensional control variables for movement primitives. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 1511–1518. IEEE (2015a).
- Rueckert, E., Mundo, J., Paraschos, A., Peters, J., and Neumann, G. Extracting Low-Dimensional control variables for movement primitives. In *Int. Conf. on Robotics and Automation* (2015b).
- Salini, J., Barthélemy, S., Bidaud, P., and Padois, V. Whole-Body motion synthesis with LQP-Based controller – application to icub. In *Modeling, Simulation and Optimization of Bipedal Walking*. Springer Berlin Heidelberg (2013).
- Salini, J., Padois, V., and Bidaud, P. Synthesis of complex humanoid whole-body behavior: a focus on sequencing and tasks transitions. In *International Conference on Robotics and Automation (ICRA)* (2011).
- Schaal, S., Mohajerian, P., and Ijspeert, A. Dynamics systems vs. optimal control — a unifying view. In T. D. Paul Cisek and J. F. Kalaska, editors, *Computational Neuroscience: Theoretical Insights into Brain Function*, pages 425–445. Elsevier (2007).
- Schaal, S., Peters, J., Nakanishi, J., and Ijspeert, A. Control, planning, learning, and imitation with dynamic movement primitives. *Workshop, IEEE Int. Conf. on Intelligent Robots and Systems (IROS)* (2003a).

-
- Schaal, S., Peters, J., Nakanishi, J., and Ijspeert, A. Learning Movement Primitives. In *International Symposium on Robotics Research* (2003b).
- Schaal, S., Peters, J., Nakanishi, J., and Ijspeert, A. Learning movement primitives. In *Robotics Research. The Eleventh International Symposium*, Springer Tracts in Advanced Robotics, pages 561–572. Springer Berlin Heidelberg (2005).
- Sentis, L. and Khatib, O. Synthesis of Whole-Body behaviors through hierarchical control of behavioral primitives. *International Journal of Humanoid Robotics* (2005).
- Stark, H. and Woods, J. *Probability and Random Processes with Applications to Signal Processing*. 3 edn. (2001).
- Stengel, R. F. *Optimal control and estimation*. Courier Corporation (2012).
- Todorov, E. General duality between optimal control and estimation. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pages 4286–4292. IEEE (2008).
- Todorov, E. and Jordan, M. Optimal Feedback Control as a Theory of Motor Coordination. *Nature Neuroscience*, 5:1226–1235 (2002).
- Toussaint, M. Robot Trajectory Optimization using Approximate Inference. In *International Conference on Machine Learning*, (ICML) (2009).
- Toussaint, M. and Goerick, C. A bayesian view on motor control and planning. In *From Motor Learning to Interaction Learning in Robots*, pages 227–252. Springer Berlin Heidelberg (2010).
- Ude, A., Gams, A., Asfour, T., and Morimoto, J. Task-Specific Generalization of Discrete and Periodic Dynamic Movement Primitives. *Transactions in Robotics*, (5) (2010).
- Williams, M., B. and Toussaint and Storkey, A. Modelling Motion Primitives and their Timing in Biologically Executed Movements. In *Advances in Neural Information Processing Systems (NIPS)* (2007).
- Yoshikawa, T. *Foundations of robotics: analysis and control*. Mit Press (1990).



Curriculum Vitæ

Education

February 2012 - now

Ph.D. in Computer Science
Intelligent Autonomous Systems Lab,
Computer Science Department,
Technische Universitaet Darmstadt, Germany

Thesis: Robot Skill Representation, Learning and Control
with Probabilistic Movement Primitives
Advisor: Jan Peters, Gerhard Neumann

December 2010

Diploma in Engineering (5-year Degree),
Electronic and Computer Engineering
Technical University of Crete, Greece

Thesis: Monas: A Flexible Software Architecture for
Robotic Agents
Advisor: Michail G. Lagoudakis

Work Experience

*January 2011 -
January 2012*

Research Associate in the EU-funded project ROBOSKIN
Cognitive Robotics Research Centre (CRRC)
University of Wales, Newport

Distinctions and Awards

3rd place at Benchmarking Reaching Movements, Humanoids, 2013
2nd place with *Noxious-Kouretes* in SPL Open Challenge, RoboCup 2011
3rd place with *Kouretes* in SPL-Nao, RoboCup 2008
Student Scholarship, Technical University of Crete, Chania, Greece, 2006

Invited Talks

Imitation Learning with Probabilistic Movement Primitives, University of Stuttgart,
Stuttgart, Germany, 2014
Probabilistic Movement Primitives, Italian Institute of Technology (IIT), Genova, Italy, 2013

Student Supervision

- Spring 2016 :* Pascal Klink,
Model Learning for Probabilistic Movement Primitives,
Bachelor Thesis, TU-Darmstadt, Germany.
- Fall 2014 - Spring 2015:* Johannes Geisler and Emmanuel Stapf,
Perceptual Coupling for Motor Primitives,
Robot Learning Project, TU-Darmstadt, Germany.
- Spring 2013 - 2014:* H. Abdulsamad, T. Buchholz, T. Croon, and M. El Khoury,
Playing Tetherball with Compliant Robots,
Integrated Project, TU-Darmstadt, Germany.
- Fall 2012 - Fall 2014:* Dimitar Ho and Viktor Kisner,
Trajectory Tracking Controller for Compliant Arm,
Integrated Project, TU-Darmstadt, Germany.
- Fall 2013 - Spring 2014:* Johannes Ringwald, *co-supervision with Gerhard Neumann*,
Combination of Movement Primitives for Robotics,
Master Thesis, TU-Darmstadt, Germany.
- Spring 2013:* Andreas Wieland and David Hoppe,
Comparison of Different Learning Algorithms for Beer-Pong,
Robot Learning Project, TU-Darmstadt, Germany.
- Fall 2012:* Johann Isaak and Manuel Kroenig,
Robot Learning Project: Beer-Pong,
Robot Learning Project, TU-Darmstadt, Germany.

Teaching Experience

- Fall 2014* Teaching Assistant, Robot Learning Lecture,
TU-Darmstadt, Germany. (Instructor: Gerhard Neumann)
- Spring 2014* Teaching Assistant, Machine Learning Lecture,
TU-Darmstadt, Germany. (Instructor: Jan Peters)
- Fall 2013* Teaching Assistant, Robot Learning Lecture,
TU-Darmstadt, Germany. (Instructor: Gerhard Neumann)
- Fall 2011* Teaching Assistant, Introduction to Artificial Intelligence,
University of Wales, Newport. (Instructor: Torbjørn Dahl)

Reviewing Activities

- Reviewer, Autonomous Robots Journal, Springer, 2014
- Reviewer, Neural Information Processing Systems Conference (NIPS), 2014-2015
- Reviewer, IEEE International Conference on Robotics and Automation (ICRA), 2014-2017
- Reviewer, IEEE International Conference on Humanoid Robots (Humanoids), 2015-2016
- Reviewer, International Conference on Intelligent Robots and Systems (IROS), 2012-2016
- Program Committee, International Joint Conference on Artificial Intelligence (IJCAI), 2013
- Reviewer, IEEE Conference on Decision and Control (CDC), 2014
- Reviewer, Int. Conference on Developmental and Learning on Epigenetic Robotics, 2014