Programming by Demonstration on Riemannian Manifolds

M.J.A. Zeestraten

Programming by Demonstration on Riemannian Manifolds

Thesis

submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy
Department of Informatics, Bioengineering, Robotics, System Engineering DIBRIS
University of Genova
and
Department of Advanced Robotics Istituto Italiano di Tecnologia

by

M.J.A. Zeestraten

Keywords: Programming by Demonstration, Riemannian Geometry, Statistics, Synergetic Control, Context Adaptation

Abstract

This thesis presents a Riemannian approach to Programming by Demonstration (PbD). It generalizes an existing PbD method from Euclidean manifolds to Riemannian manifolds. In this abstract, we review the objectives, methods and contributions of the presented approach.

Objectives

PbD aims at providing a user-friendly method for skill transfer between human and robot. It enables a user to teach a robot new tasks using few demonstrations. In order to surpass simple record-and-replay, methods for PbD need to 'understand' *what to imitate*; they need to extract the functional goals of a task from the demonstration data. This is typically achieved through the application of statistical methods.

The variety of data encountered in robotics is large. Typical manipulation tasks involve position, orientation, stiffness, force and torque data. These data are not solely Euclidean. Instead, they originate from a variety of manifolds, curved spaces that are only locally Euclidean. Elementary operations, such as summation, are not defined on manifolds. Consequently, standard statistical methods are not well suited to analyze demonstration data that originate from non-Euclidean manifolds. In order to effectively extract *what-to-imitate*, methods for PbD should take into account the underlying geometry of the demonstration manifold; they should be *geometry-aware*.

Successful task execution does not solely depend on the control of individual task variables. By controlling variables individually, a task might fail when one is perturbed and the others do not respond. Task execution also relies on couplings among task variables. These couplings describe functional relations which are often called synergies. In order to understand *what-to-imitate*, PbD methods should be able to extract and encode synergies; they should be *synergetic*.

In unstructured environments, it is unlikely that tasks are found in the same scenario twice. The circumstances under which a task is executed—the task context—are more likely to differ each time it is executed. Task context does not only vary during task execution, it also varies while learning and recognizing tasks. To be effective, a robot should be able to learn, recognize and synthesize skills in a variety of familiar and unfamiliar contexts; this can be achieved when its skill representation is *context-adaptive*.

The Riemannian Approach

In this thesis, we present a skill representation that is *geometry-aware*, *synergetic* and *context-adaptive*. The presented method is probabilistic; it assumes that demonstrations are samples from an unknown probability distribution. This distribution is approximated using a Riemannian Gaussian Mixture Model (GMM).

Instead of using the 'standard' Euclidean Gaussian, we rely on the Riemannian Gaussian—a distribution akin the Gaussian, but defined on a Riemannian manifold. A Riemannian manifold is a manifold—a curved space which is locally Euclidean—that provides a notion of distance. This notion is essential for statistical methods as such methods rely on a distance measure. Examples of Riemannian manifolds in robotics are: the Euclidean space which is used for spatial data, forces or torques; the spherical manifolds, which can be used for orientation data defined as unit quaternions; and Symmetric Positive Definite (SPD) manifolds, which can be used to represent stiffness and manipulability.

The Riemannian Gaussian is intrinsically *geometry-aware*. Its definition is based on the geometry of the manifold, and therefore takes into account the manifold curvature. In robotics, the manifold structure is often known beforehand. In the case of PbD, it follows from the structure of the demonstration data. Like the Gaussian distribution, the Riemannian Gaussian is defined by a mean and covariance. The covariance describes the variance and correlation among the state variables. These can be interpreted as local functional couplings among state variables: synergies. This makes the Riemannian Gaussian synergetic. Furthermore, information encoded in multiple Riemannian Gaussians can be fused using the Riemannian product of Gaussians. This feature allows us to construct a probabilistic context-adaptive task representation.

CONTRIBUTIONS

In particular, this thesis presents a generalization of existing methods of PbD, namely GMM-GMR and TP-GMM. This generalization involves the definition of Maximum Likelihood Estimate (MLE), Gaussian conditioning and Gaussian product for the Riemannian Gaussian, and the definition of Expectation Maximization (EM) and Gaussian Mixture Regression (GMR) for the Riemannian GMM. In this generalization, we contributed by proposing to use parallel transport for Gaussian conditioning. Furthermore, we presented a unified approach to solve the aforementioned operations using a Gauss-Newton algorithm. We demonstrated how synergies, encoded in a Riemannian Gaussian, can be transformed into synergetic control policies using standard methods for Linear Quadratic Regulator (LQR). This is achieved by formulating the LQR problem in a (Euclidean) tangent space of the Riemannian manifold. Finally, we demonstrated how the context-adaptive Task-Parameterized Gaussian Mixture Model (TP-GMM) can be used for context inference—the ability to extract context from demonstration data of known tasks. Our approach is the first attempt of context inference in the light of TP-GMM. Although effective, we showed that it requires further improvements in terms of speed and reliability.

The efficacy of the Riemannian approach is demonstrated in a variety of scenarios. In shared control, the Riemannian Gaussian is used to represent control intentions of a human operator and an assistive system. Doing so, the properties of the Gaussian can be employed to mix their control intentions. This yields shared-control systems that continuously re-evaluate and assign control authority based on input confidence. The context-adaptive TP-GMM is demonstrated in a Pick & Place task with changing pick and place locations, a box-taping task with changing box sizes, and a trajectory tracking task typically found in industry.

PREFACE

I first encountered Programming by Demonstration (PbD) when I visited Dr. Sylvain Calinon at Istituto Italiano di Tecnologia (IIT) as a master student in 2012. Around that time, I was also introduced to the world of geometry through the course 'Modern Robotics' given at TU Delft by Prof. Stefano Stramigioli. This sparked the idea of using geometrical concepts to enhance the abilities of PbD.

When I started my Ph.D adventure at IIT in September 2015, the idea still inspired me, but I lacked the understanding of geometrical concepts to put two and two together. It was only after the DISC summer school 2016 (Zandvoort, the Netherlands) that the Riemannian approach would take shape. There, I met Alessandro Saccon with whom I discussed the concept of statistics on manifolds. He portrayed this topic with the Roman saying:"hic abundant leones"; but motivated me to pursue this direction (as I was still young). The challenge to defeat the lions, motivated me to unwrap the world of Riemannian geometry, which led to this dissertation.

This brief chronicle shows that ideas are like Italian wine: they require time to ripe. It also highlights that research is not a single-man journey; it relies on exchange and open discussion of ideas. Although I compiled this thesis on my own, I did not shape its ideas in solitude. Therefore, I find it inappropriate, if not selfish, to write this thesis in first person singular. Instead, we will continue in first person plural. You can interpret 'we' like 'you and I', as we move through the argument, or as the group of people who made it come to light.

But before proceeding, I want to acknowledge the 'we' for enabling me to accomplish this thesis. First of all, I owe great gratitude to Sylvain for his supervision, faith, support, time, corrections, and frequent visits to IIT. Without him, this PhD-journey would not have started, progressed or finished. I want to thank Prof. Caldwell for providing me the opportunity to perform research at IIT, and his support throughout the Ph.D process. Furthermore, I want to thank Danilo for his mathematical insights, Leonel for his encouraging words in time of need, and João and Ioannis for their valuable input during discussions on the topic of this thesis. And, of course, Milad, Brian, Domingo, Yan Long and Fares for their valuable input during group meetings. Additionally, I enjoyed working with the AutoMAP team and thank Fei, Boyan, Arrfou, Andrea, and Francesco for providing me the opportunity to work with them. Finally, I want to thank Prof. Dr. Torras and Prof. Dr. Steil for reviewing this thesis, and providing helpful feedback.

As a Marie-Curie early-stage researcher, I was privileged to work internationally with a group of amazing Ph.D students, post-docs and professors. I especially want to thank Aaron, Andrea, Esra and Matthias for hosting me at the technical university of Munich. Furthermore, I want to thank the ADVR secretaries—Silvia in particular— for handling all administration that came along with the Marie-Curie project.

The research environment at IIT makes you get to know an amazing amount of people who shaped my social life in Genova. I owe gratitude to all off them. In particular,

I want to thank Stefano, Nawid, Jörn, Arren, Trina, Wesley, Renske, Prezemek, Matej, Navvab, Arash, Claudio and Steve for making time for coffee, beer, lunch, dinner, or squash.

I want to thank my family; my parents, sisters and brother. Firstly, for their support. Secondly, for their friendly-understanding nod, which followed my void when they asked me: What are you exactly doing at IIT? Furthermore, I want to thank my parents in law for motivating me to pursue a Ph.D.

But most gratitude I owe to my significant other, Jana. Gratitude for leaving Delft and moving with me to Genova; gratitude for her unprecedented and unconditional support, patience and love; gratitude for giving me Viko, our son, who always lifts my spirit.

Martijn Zeestraten
Genova, November 2017

Notation & Illustration Guide

Notation guidelines

This thesis mainly follows the notation guidelines stated in the following table.

Notation	Description	Meaning
a	Normal case	Scalar value
$\boldsymbol{a}$	Bold lower-case	Manifold element
$\mathtt{a}$	Type writer style	Random variable
$\mathfrak{a}$	Math fraktur	Tangent space element
A	Upper-case	Set
$\boldsymbol{A}$	Bold upper-case	Matrix
$\boldsymbol{A}^\top$	Superscript script-style T	Matrix transpose
$\tilde{a}$	Over-placed tilde	Approximated value
a_i	Subscript letter	Index indicator (typically i, j, k or p)
a^c	Superscript letter	Functional indicator (e.g. $\boldsymbol{\theta}^c$ for context parameters)
$\mathfrak{a}_{\boldsymbol{g}}$	Bold subscript letter	Relation indicator (here meaning $\mathfrak{a} \in \mathcal{T}_{\boldsymbol{g}}\mathcal{M}$)
$\boldsymbol{A}_{\boldsymbol{g}}^{\boldsymbol{h}}$	Bold sub- and super-scripts	Transformation from $\boldsymbol{g}$ to $\boldsymbol{h}$

Occasionally, the guidelines are breached; if so the notation is clarified within the text.

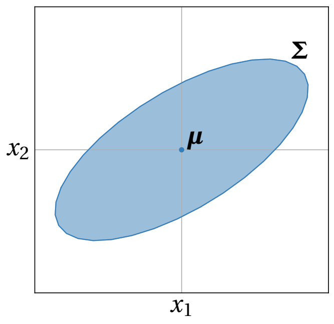
COMMON SYMBOLS

The following is a list of commonly used symbols.

Symbol	**Description**
$\mathcal{A}_{\boldsymbol{g}}^{\boldsymbol{h}}(\boldsymbol{p})$	Action function (see Section 2.2.1)
$\mathcal{A}_{\parallel\boldsymbol{g}}^{\boldsymbol{h}}(\mathfrak{p})$	Parallel action function (see Section 2.2.1)
det()	Function which returns the determinant of a matrix
diag()	Function which transforms a vector into diagonal matrix, or extracts diagonal components from a matrix.
$\mathrm{Exp}_{\boldsymbol{p}}()$	Riemannian exponential map, defined at $\boldsymbol{p}$ (see Section 2.2.1)
$\boldsymbol{I}$, $\boldsymbol{I}_d$	Identity matrix (of dimension d).
$\boldsymbol{J}$	Jacobian
K	Number of Gaussians in GMM
$\boldsymbol{\Lambda}$	Precision matrix
λ	Precision scalar
$\mathrm{Log}_{\boldsymbol{p}}()$	Riemannian logarithmic map, defined at $\boldsymbol{p}$ (see Section 2.2.1)
$\boldsymbol{\mu}$	Mean of Gaussian distribution
$\mathcal{M}$	Riemannian manifold
$\mathcal{N}()$	Gaussian distribution
N	Number of data points
$\boldsymbol{\omega}$	Vector of angular velocities
π	The mathematical constant pi
π_i	Mixing coefficient of Gaussian i in a GMM
P	Number of coordinate systems in a TP-GMM
ρ	Correlation coefficient
$\mathbb{R}^d$	d-Dimensional Euclidean space, or d-dimensional set of real numbers
$\mathbb{R}^+$	Set of positive real numbers
$\boldsymbol{R}$	Rotation matrix (Chapters 2 and 4), or control cost matrix (Chapters 3 and 5)
SE(d)	d-Dimensional special Euclidean Group
SO(d)	d-Dimensional special orthogonal Group
σ	Variance scalar
$\boldsymbol{\Sigma}$	Covariance matrix of Gaussian distribution
$\mathcal{S}^d$	Spherical manifold of dimension d
$\mathcal{S}^+$	Manifold of SPD matrices
$\boldsymbol{\theta}^m$, $\boldsymbol{\theta}^c$	Model and context parameters, respectively
$\mathcal{T}_{\boldsymbol{p}}\mathcal{M}$	Tangent space of manifold $\mathcal{M}$ defined at $\boldsymbol{p} \in \mathcal{M}$

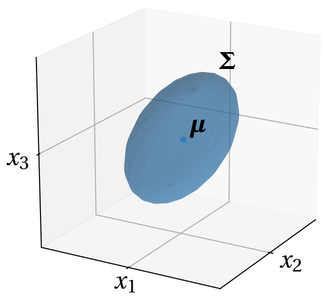
ILLUSTRATION GUIDE

2D Euclidean Gaussian



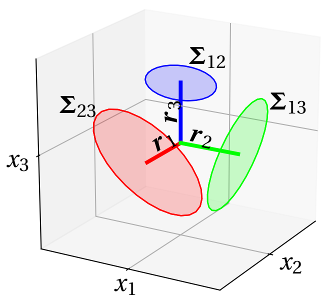
A Euclidean Gaussian is characterized by its mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. For 2D Gaussian distributions, these are visualized using a point and an ellipse, respectively. The ellipse contour is drawn one standard deviation from its center (the mean $\boldsymbol{\mu}$), unless stated otherwise. Along the contour of the ellipse, the distribution has a constant probability, as a result its shape corresponds to the covariance $\boldsymbol{\Sigma}$.

3D Euclidean Gaussian



A Euclidean Gaussian is characterized by its mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. For 3D Gaussian distributions, the covariance is visualized using an ellipsoid, and the mean is sometimes not depicted as it lies at the ellipsoid center. The ellipsoid surface is drawn one standard deviation from the mean, unless stated otherwise. Over the surface of the ellipsoid, the distribution has a constant probability, as a result its shape corresponds to the covariance of the distribution.

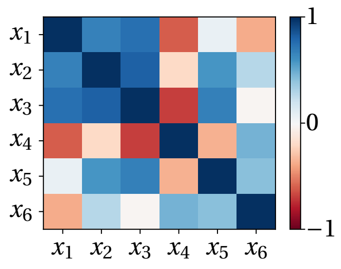
Gaussian 3D Orientation



$\boldsymbol{\Sigma}_{12}$
$\boldsymbol{\Sigma}_{23}$
$\boldsymbol{\Sigma}_{13}$
$\boldsymbol{r}_3$
$\boldsymbol{r}_1$
$\boldsymbol{r}_2$
x_3
x_1
x_2

A Gaussian distribution on 3D orientation is visualized using three axes $(\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3)$, colored in red, green and blue, with ellipses at their end-points. The axes depict the mean orientation, and represent the (scaled) columns of the corresponding rotation matrix. Their origin usually depicts the spatial mean. The ellipse contours show the axes end-points motion along a line of constant probability; they display the orientation covariance at one standard deviation (unless stated otherwise). The combination of the marginal covariances, $\boldsymbol{\Sigma}_{12}$, $\boldsymbol{\Sigma}_{13}$, $\boldsymbol{\Sigma}_{23}$, cover the full orientation covariance.

Correlation Matrix



The entries of a covariance matrix, $\Sigma_{ij} = \rho_{ij}\sigma_i\sigma_j$, combine correlation coefficients $-1 \leq \rho_{ij} \leq 1$ among random variables x_i and x_j, with their standard deviations σ_i, σ_j. The correlation matrix visualizes couplings among state variables (synergies). Its cell colors correspond to the values of the correlation coefficients. As diagonal cells represent the correlations of variables with themselves, their value is always 1.

Contents

1

INTRODUCTION

Essentially, robots are mechatronic systems; assemblies of well-engineered sensors, actuators and structural components. Opposed to most mechatronic systems, robots are envisioned to perform a variety of tasks in unstructured environments. Thereby achieving or surpassing human performance in terms of speed, accuracy and adaptability. Achieving this vision is challenging. The task variety and lack of structure makes manual coding of control policies impractical, if not unfeasible. Robot learning provides a promising alternative. Instead of manually coding the control policy, the robot derives it from examples, or discovers it through reward-driven exploration. In this work, we focus on the former: Programming by Demonstration (PbD) [1–4], and propose to use Riemannian geometry to model, synthesize and recognize task space skills.

1.1. BACKGROUND

Although early works in artificial intelligence predicted autonomous robots would emerge before the year 2000, history demonstrated differently [5]. Classical approaches in robot learning (e.g. Reinforcement Learning (RL) [6]), learn discrete state-action pairs from scratch. Unfortunately, these approaches do not scale well to the robots's continuous state-action space due to *the curse-of-dimensionality* [6].

To mitigate this problem, researchers proposed to pre-structure the learning problem using biologically inspired *movement primitives* [7, 8]. These are elementary units of action that can be adapted, merged and sequenced into complex behavior [9]. Examples of such primitive movements are *pick-up object A, kick the ball, wave to person A*. Instead of using atomic state-action pairs, movement primitives use parameterized functions. These functions replace the discrete state-action pairs using a continuous state-action relation. Typical approaches for parameter estimation are PbD [1–4] and policy search [6, 10, 11]. Often, these are combined: initializing the parameters of the primitive based on human demonstration, and improving the primitive over time using exploration strategies.

Movement primitives can be defined at different levels. Two commonly encountered levels are joint-space and task-space [7]. Task-space primitives can be more generic, as they model primitives independently from a kinematic structure. This eases skill transfer between dissimilar agents, and allows the robot to use its redundancies to minimize influence of sensory-motor noise [12, 13], or to perform a secondary task [14]. Additionally, the encoded task coordination enables the robot to maintain task performance while being perturbed [15, 16] (task-specific flexibility).

1.2. MOTIVATION & OBJECTIVES

Conceptually, PbD allows the transfer of skill through physical demonstrations. Yet, to surpass simple record-and-replay abilities, the robot needs to 'understand' the objective which underlies the demonstrations; i.e. *what to imitate?* Furthermore, given the objective is uncovered, the robot needs to know how to use its body to achieve it, i.e. *how to imitate?* These questions are among the four elementary questions of PbD [2], and underlie the objectives of the proposed skill representation, namely: geometry-aware, synergetic and context-adaptive. In what follows, we motivate why we focus on these objectives.

1.2.1. GEOMETRY-AWARE

In robotics, we encounter a variety of manifolds. For example in task space, we find the Euclidean manifold $\mathbb{R}^d$ to describe positions, forces and torques; the manifold of symmetric positive definite matrices $\mathcal{S}^+$ to represent stiffness, covariance or manipulability; the special orthogonal group SO(3) or the unit-sphere $\mathcal{S}^3$ to represent orientation by rotation matrices or unit quaternions; or the special Euclidean group SE(3) to describe rigid-body poses.

Demonstration data can originate from combinations of these manifolds. In order to uncover the task objective—to answer *what-to-imitate?*—a skill representation needs to admit data from different manifolds in order to fully benefit from the available information. By combining information from a variety of manifolds, the objective could be uncovered more effectively or accurately. Furthermore, to synthesize the learned behavior, the robot is restricted to the geometry of the output space. If the desired output is an orientation, the algorithm should ensure the output is a unit quaternion, or an orthonormal rotation matrix. Therefore, the skill parameterization should be *geometry-aware.*

1.2.2. SYNERGETIC

Successful task execution does not solely depend on the control of individual task variables. By controlling variables individually, a task might fail when one is perturbed and the others do not respond. Instead, successful task execution depends on coupling of task variables. These coupling describe functional relations which are often called *synergies* [16, 17].

In control, synergies can be used to synthesize (reflexive) control policies [18, 19] using the minimal intervention principle [12]. Such policies maintain functional integrity of the encoded skill, opposed to tracking the individual state variables. Consider for example a robot which holds a tray with two hands. Its objective is to keep the distance between the hands equal. When the robot is pushed on the left hand, a synergetic controller would respond compliantly, moving both hands in concord. Yet, when the right hand is fixed at time of perturbation, the left hand should resist the perturbation. As the synergetic controller aims at maintaining functional integrity at minimum cost, different disturbance scenarios are handled differently. In contrast, a control policy that ignores coupling among task variables could either be stiff or compliant and propagate the interaction forces from the left to the right hand through the tray.

A skill encoding should describe the coupling and variation of the task variables. Besides control, this information also serves regression, generalization and control, as demonstrated by Mühlig *et al.* [20] Calinon *et al.* [18, 19, 21] and Paraschos *et al.* [22, 23].

1.2.3. CONTEXT-ADAPTIVE

In unstructured environments, a task can be encountered in different contexts. To allow the robot to respond to new situations, its skill representation should be context-adaptive. Although *what-to-imitate?* intrinsically implies the capacity to generalize, we explicitly mention context adaptation to stress its importance.

The ability to learn, synthesize and recognize tasks executed in different contexts makes a robot more versatile and user-friendly. Skill adaptation increases versatility, as it allows the robot to perform tasks in contexts that were not encountered before. The ability to learn a task from samples performed in different context, increases the ease of transfer: it lessens the need to anticipate on future context; and it reduces the number of models to train (one task model for all context, instead of a task model for each context); it removes the need to replicate the same context for separate demonstrations.

1.3. STATE OF THE ART

The presented objectives have been recognized by others, and existing approaches include them to some extent. Table 1.1 compares state of the art primitive parameterizations. It shows that that none of the existing methods fully meet all criteria. The remainder of this section motivates this conclusion by discussing the table content in detail.

Method	Adaptive	Synergetic	Geometry-Aware	Related Work
DMP	✓		✓	[8, 24–31]
TP-GMM	✓	✓		[18, 19, 21, 32–36]
ProMP	✓	✓		[22, 23]
GPR	✓	✓*	✓*	[37–39]

Table 1.1: Comparison of common primitive representations based on the objectives presented in this chapter. *) Gaussian Process Regression (GPR) can encode coordination among state variables separately using a Wishart process [39], and is geometric aware on the input only [38].

1.3.1. DMP

Dynamic Movement Primitives (DMP) [8, 24] are widely used to represent motion primitives. A DMP consists of a spring-damper system that is perturbed by non-linear forcing term. The non-linear forcing term shapes the behavior of the primitive, and its parameters are learned from data. The linear spring-damper ensures that the DMP converges to a pre-defined goal position. A DMP is context-adaptive through its goal position, which can be adapted (online) to new goal positions. However, additional motion constraints, such as approach direction, are not encoded by DMP.

Generally, each state dimension is controlled by a separate DMP. Although these DMPs are temporally coupled through a common phase variable, they are not spatially coupled. Furthermore, the output of the forcing terms does not depend on the system state, which makes their behavior mostly open-loop. The absence of feedback and spatial coupling prevents synergetic encoding. Ude *et al.* [26] used geometrical concepts to encode end-effector pose using DMPs. This extension thus made DMP geometry-aware.

1.3.2. GMM-GMR AND TP-GMM

Calinon *et al.* [21, 32–35] approach skill encoding as a clustering problem. Instead of modeling the regression function directly, they estimate a joint-distribution over the state variables, and perform regression through conditioning. This approach allows the handling of missing data, and the encoding of variance and coupling among state variables. During reproduction, the motion is retrieved through statistical inference using Gaussian Mixture Regression (GMR). This approach, named GMM-GMR, has been used to represent movement skills using dynamical systems [34,35] and time (or phase) driven systems [21, 32, 33].

The regression output of GMR is a Gaussian distribution with full covariance matrix. The variance and correlation encoded in this matrix describe the local synergies in the demonstration data. Mühlig *et al.* [20] used the variance resulting from GMR to optimize movement primitives. Later works, exploit the full covariance information to obtain synergetic response to perturbations [18, 19, 36].

The linear transformation properties of the Gaussian Mixture Model (GMM) allow the modeling of primitives in different coordinate systems. By relating coordinate systems to the task context (e.g. objects or landmarks), a context-adaptive version of GMM-GMR, named Task-Parameterized Gaussian Mixture Model (TP-GMM), is realized [18,21, 40].

The Euclidean nature of the Gaussian distribution, restricts the type of data that can be used with GMM-GMR and makes it not geometry-aware. Despite these limitations, Silvério *et al.* [41] were able to extend TP-GMM for unit quaternions by exploiting the linear representation of the quaternion product. As this approach omits the curvature of the unit quaternion manifold, it requires normalization of the quaternions after regression and lacks geometrical interpretation of the covariance information. Kim *et al.* [42] presented a more geometric approach to represent orientation in GMM-GMR. Although similar to the approach presented in this thesis, Kim *et al.* do not use *parallel transport*, an essential element for the generalization of Gaussian conditioning as will be demonstrated in Section 2.5.4.

1.3.3. PROMP

Probabilistic Movement Primitives (ProMP) [22, 23] represent primitives using parameterized trajectory distributions. The structure of ProMP encodes both temporal and spatial couplings among the state variables. These couplings facilitate context-adaptation and synergy encoding. Using conditioning, the behavior of a ProMP can be adapted to move through via-points at specific time instances. The spatial and temporal couplings ensure that the motion characteristics are maintained.

The structure of ProMP consists of a weighted sum of basis functions, and splits spatial and temporal information. This structure allows the construction of closed-form synergetic controllers for linear systems, and efficient motion adaption through Gaussian conditioning. Yet, as the structure relies on Euclidean operations, ProMP is restricted to Euclidean data. Furthermore, because ProMP models a distribution over trajectories, each demonstration only represents a single data point. This, in combination with the large number of open parameters (mainly due to the size of the covariance matrix), makes the training of ProMP require a large number of demonstrations [43].

1.3.4. GPR

GPR is a generic non-parametric regression technique which can be used for time-series modeling [37, 44]. Unlike GMM-GMR and ProMP, it does not capture variability but uncertainty. This implies the following: if many (Gaussian) samples are available for a state at time t, a GPR will output a desired state value with high certainty, even if the variance of the samples is large. GMM-GMR and ProMP will instead output the same mean value, but with a variance corresponding to the observed samples. In addition, a separate GPR is required for each task variable. As a result, GPR cannot directly encode coupling among the task variables. Recently, Umlauft *et al.* [39] proposed to capture coupling and variability separately using a Wishart distribution (a Gaussian process on Symmetric Positive Definite (SPD) matrices). Using this route, GPR is able to encode synergies using a seperate Gaussian Process.

In robotics, GPR is typically used with a radial-basis kernel as prior. Using this kernel GPR is only compatible with Euclidean data. This because both input and output depend on Euclidean concepts: the input because the radial-basis function relies on a Euclidean distance measure; the output because the individual GPR outputs are unconstrained and therefore can only be combined into a Euclidean vector. Lang *et al.* [38] remove the input limitation using an alternative distance measure in the radial-basis kernel. Using this kernel, the input the GPR becomes geometry-aware.

1.4. A RIEMANNIAN APPROACH

This thesis proposes a way to extend TP-GMM to Riemannian manifolds, and thus generally applicable to a wider range of demonstration data. This generalization allows us to exploit context-adaptive structure of TP-GMM in task-space manipulation, and use its synergetic properties. This proposed extension checks the 'Geometry-Aware' cell of TP-GMM in Table 1.1, and makes TP-GMM meet the desired objectives.

We start by motivating the use of Riemannian manifolds; why does a manifold need to be Riemannian in order to generalize TP-GMM? The answer lies in the requirements of TP-GMM. Its working relies on the Gaussian distribution and its properties: TP-GMM requires the manifold to admit a probability distribution. This, in turn, requires the notion of (Mahanalobis) distance. Then, in order to attain TP-GMM functionality, the distribution should have equivalent operations for Maximum Likelihood Estimate (MLE), Gaussian conditioning, Gaussian product, and linear transformation.

The choice for Riemannian manifolds follows from these requirements. Informally stated, manifolds are smooth curved spaces that locally resemble a Euclidean space. Generally, manifolds do not provide a notion of distance, as they do not possess a metric to measure distance. Riemannian manifolds provide this notion, using the Riemannian metric—a positive definite inner product. Note, that this approach explicitly leverages the structure of the manifold which is known beforehand. This contrasts with the field of *metric learning*, where researchers attempt to discover the structure of the manifold from data [45, 46].

With the notion of distance, one can define Riemannian equivalents of mean and covariance, and generalize the Gaussian distribution as demonstrated by Pennec [47]. We follow this work, and rely on the information-based generalization of the Gaussian, which we denote the *Riemannian Gaussian*. The Riemannian Gaussian permits all properties required to generalize GMR and TP-GMM to Riemannian manifolds. The proposed generalization of TP-GMM maintains its context-adaptive and synergetic properties, and gives it the ability to consider a wider range of demonstration data. In this thesis, we focus on the use of position and orientation data, but others build on the proposed approach [48] and demonstrated its use on other manifolds [49, 50].

1.5. CONTRIBUTIONS

The main contribution of this thesis is the generalization of TP-GMM and related methods to Riemannian manifolds. This generalization required several innovative steps, which will be presented below. We split the contributions according to the presented objectives: geometry-aware, synergetic and context-adaptive. Detailed descriptions of the contributions are given in the introduction of the corresponding chapters.

1.5.1. GEOMETRY-AWARE

The generalization of GMM-GMR from Euclidean manifolds to Riemannian manifolds, involved generalizing the GMM and its operations, namely: parameter estimation, Gaussian product, Gaussian conditioning and linear transformation. Compared to previous work [42, 51], we propose to use *parallel transport* in Gaussian conditioning and GMR, and demonstrate its necessity. MLE has been proposed previously (e.g. MLE [47, 51, 52]), and product of Riemannian Gaussians appeared as fusion in other fields [53–56]. Yet in this thesis, we provide a unified approach to all operations, and show their applicability in PbD.

1.5.2. SYNERGETIC

The Riemannian Gaussian encodes variation and correlation among the manifold dimensions. This information represents functional grouping of the manifold dimensions; it describes (local) synergies. Previous work demonstrated how standard Linear Quadratic Regulator (LQR) can be used to obtain synergetic, or risk-sensitive, state-feedback controllers based on the encoded covariance [18, 19, 36]. This thesis generalizes these concepts to Riemannian manifolds. Although generally LQR cannot be directly applied on manifolds, we demonstrate that such controllers can be defined using the linear tangent spaces of the Riemannian manifold.

1.5.3. Context-Adaptive

Our Riemannian approach, generalizes the context-adaptive properties of TP-GMM to task space. To date, TP-GMM applications have restricted themselves to rigid-body transformations (rotations and translations of the demonstrated motions). Yet, the transformation properties of the Gaussian permit the broader range of affine transformations. In this thesis, we formalize this range, and demonstrate the capabilities of TP-GMM with affine context parameterizations.

TP-GMM is typically applied to model and synthesize robot skills in a context-adaptive manner. The introduction of task context brings about a third action: context inference, the estimation of circumstances under which a skill has been executed. In the light of TP-GMM, we address this problem for the first time. The proposed method uses an Expectation Maximization (EM)-based algorithm to infer context from movement data given a context-adaptive model.

1.5.4. Applications

The research that led to this thesis has been performed within the SMART-E project [57]. The focus of this project lies on robotics research with applications in industrial scenarios. In this perspective, the Riemannian approach for PbD has been applied in practical scenarios.

The main application involved maintenance of the Large Hadron Collider (LHC) of CERN. As the LHC is highly radioactive, humans cannot enter it to perform maintenance. Instead, CERN relies on teleoperation to perform maintenance operations. In this thesis we explore ways in which the Riemannian framework can be used to generate shared control strategies that can assist the teleoperator. Within the AutoMAP project [58], the Riemannian framework has also been used to program autonomous maintenance operations in a similar scenario.

1.6. Thesis outline

The contributions are split over the main objectives of the thesis: geometry-aware, synergetic, and context-adaptive, as illustrated in Figure 1.1. Each chapter details how the Riemannian approach achieves an objective. Chapter 2 describes the theoretical foundation of the Riemannian approach to PbD. Chapter 3 describes how the Riemannian Gaussian can be used to synthesize synergetic controllers. Chapter 4 describes how context adaptation is achieved using the Riemannian framework: the generalization of TP-GMM. Chapter 5 combines elements of previous chapters to develop two different shared control strategies. The chapters are mostly self-contained, as they comprise a dedicated introduction, related work and discussion sections. Conclusions and future work are given in Chapter 6.

1.7. Supplementary material

The methods described in this thesis have been implemented and bundled in a Python module named RiePybdlib. The module can be found via the author's website: `https://gitlab.martijnzeestraten.nl/martijn/riepybdlib`. The website also provides tutorial articles, in the form of interactive notebooks that demonstrate the operations

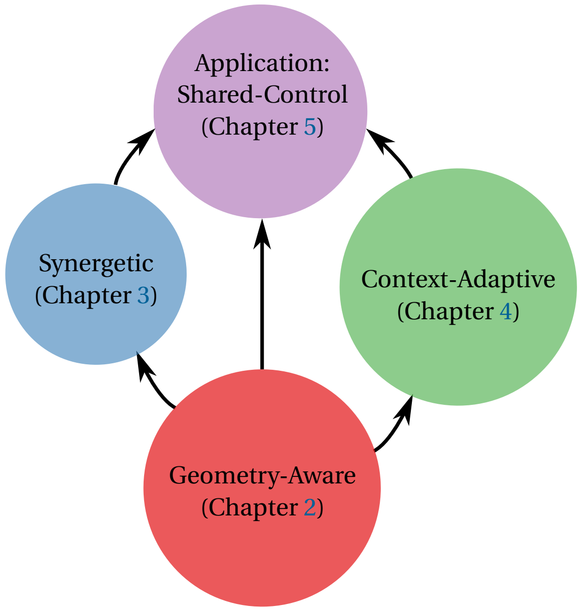

Application: Shared-Control (Chapter 5)
Synergetic (Chapter 3)
Context-Adaptive (Chapter 4)
Geometry-Aware (Chapter 2)

Figure 1.1: Graphical representation of the thesis contents.

presented in Chapter 2 and Riemannian LQR presented in Chapter 3. Matlab code has been made available by Dr. Sylvain Calinon and can be accessed via: `http://www.idiap.ch/software/pbdlib/`.

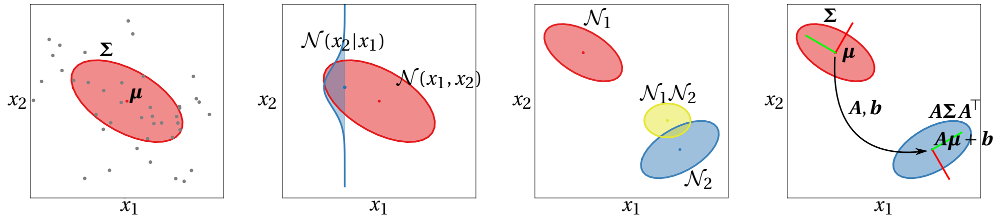
Videos of the experiments presented in chapters 2, 3 and 4 can be found via the following links:

- Chapter 2: `https://youtu.be/NiRPEOegymk`,
- Chapter 3: `https://youtu.be/oM5btdbsdig`,
- Chapter 4: `https://youtu.be/jYas1LZAtMI`.

2

A Geometry-Aware Task Encoding

2.1. Introduction



$\boldsymbol{\Sigma}$
$\boldsymbol{\mu}$
x_2
x_1
$\mathcal{N}(x_2|x_1)$
$\mathcal{N}(x_1,x_2)$
$\mathcal{N}_1$
$\mathcal{N}_1\mathcal{N}_2$
$\mathcal{N}_2$
$\boldsymbol{A},\boldsymbol{b}$
$\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^\top$
$\boldsymbol{A}\boldsymbol{\mu}+\boldsymbol{b}$

(a) Maximum Entropy (b) Conditioning (c) Product (d) Linear Transformation

Figure 2.1: Visualization of the favorable properties of the Gaussian that are often used in PbD. (a) Given the first two moments ($\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$) the Gaussian is the maximum entropy distribution. (b) The conditional distribution of a jointly Gaussian distribution is again Gaussian. (c) The product of two Gaussians is, after normalization, a Gaussian distribution. (d) The linear transformation of a Gaussian is again Gaussian.

Probabilistic methods in Programming by Demonstration (PbD) assume demonstrations represent samples from some unknown probability distribution. The models used to approximate this distribution are often based on the Gaussian distribution, e.g. [8, 21, 22, 59, 60]. The Gaussian has various favorable properties, as illustrated in Figure 2.1: it is the maximum entropy distribution given the first two moments (mean and covariance) of the data; the conditional distribution of joint Gaussian distribution, is again Gaussian; the product of two Gaussians is, after normalization, Gaussian; and, a Gaussian remains Gaussian under linear transformation of its parameters. These properties have enabled the estimation, synthesis and generalization properties of popular approaches

Parts of this chapter have been published in IEEE Robotics and Automation Letters (RA-L) [48].

such as GMM-GMR [33], TP-GMM [21], Probabilistic Movement Primitives (ProMP) [22] and Stable Estimator of Dynamical Systems (SEDS) [35].

Common in these approaches is the encoding of movement data. Such data can be defined in joint or end-effector space. Often, the primary task requirements that underlie a movement are related to the end-effector space—commonly referred to as *task space*. In such cases, joint-level descriptions may be unnecessarily restrictive: they are limited to a single kinematic structure (a change in this structure may change the task-space result), and cannot exploit kinematic redundancy for secondary tasks. A task-space representation is more easily transferred among robots (this only requires the transformation between the robots end-effector poses), and it allows inclusion of secondary tasks in the null-space [61, 62]. Furthermore, it allows the robot to exploit redundancy to reduce the effect of noise on task performance [12, 13], to reduce expected impact in collision [63], or to effectively detect and respond to collision [64].

As orientation data cannot be globally expressed in a Euclidean space, the Gaussian distribution is not very well suited to encode orientation or pose data. This may very well explain why many probabilistic approaches either consider joint-space or position-only task-space representations: the Gaussian-based methods do not permit data defined in non-Euclidean spaces. Although normalization techniques can in some cases be used to let Euclidean methods comply with the manifold curvature, these techniques introduce inaccuracies due to the Euclidean length measurement. This is illustrated for the manifold $\mathcal{S}^1$ in Figure 2.2: although normalization can be used to project Euclidean estimates onto the manifold, the projection does not correspond to the mean computed with true manifold distances.

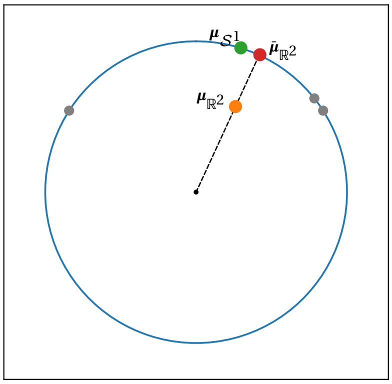

$\boldsymbol{\mu}_{\mathcal{S}^1}$
$\bar{\boldsymbol{\mu}}_{\mathbb{R}^2}$
$\boldsymbol{\mu}_{\mathbb{R}^2}$

Figure 2.2: The mean of three data points in $\mathcal{S}^1$ computed using different methods: based on manifold distance (green), based on Euclidean distance (orange), based Euclidean distance and normalized (red).

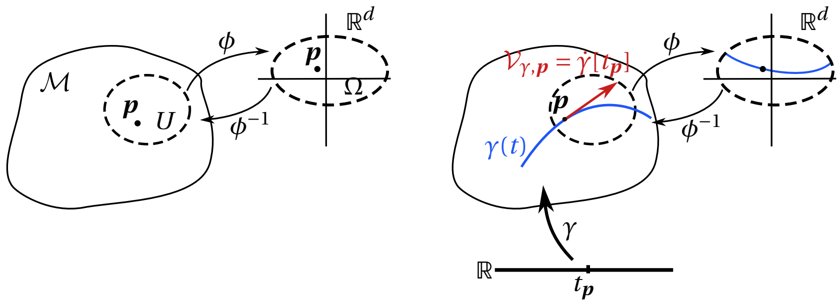
In this thesis, we propose a geometry-aware approach. Using the tools of Riemannian geometry, we obtain a task representation that can encode demonstration data originating from a variety of manifolds. The Riemannian approach allows us to generalize approaches based on Gaussian Mixture Model (GMM). Although, this was originally motivated by the ability to handle orientation data, the proposed approach can also handle other type of data such as Symmetric Positive Definite (SPD) matrices. These are encountered when handling stiffness, inertia or sensory data organized as covariance features [49, 50].

Section 2.5 presents the foundation of the Riemannian approach to PbD. It introduces Riemannian statistics and shows how to generalize the Gaussian distribution and the required properties to Riemannian manifolds. Before presenting the core material, preliminaries on Riemannian geometry and common parameterizations of orientation are discussed in Section 2.2. Furthermore, we review how others have countered statistical encoding of orientation in Section 2.3, and motivate our parameterization of end-effector pose in Section 2.4. After introduction of the Riemannian approach, Section 2.6 presents an application of it in bi-manual manipulation.

2.2. PRELIMINARIES

The approach presented in this thesis relies on Riemannian geometry. We therefore start this section by introducing the required notions from this field. More elaborate discussion on Riemannian geometry are given by Lee [65] and Jost [66]. In addition, we describe the links of various parameterizations of orientation with Riemannian geometry and assess their usability for our statistical application.

2.2.1. RIEMANNIAN MANIFOLDS



(a) Coordinate chart

(b) Tangent vector $\mathcal{V}_{\gamma,\boldsymbol{p}}$

Figure 2.3: (a): A manifold $\mathcal{M}$ is a d-dimensional topological space for which each point $\boldsymbol{p}$ has a neighborhood $U \in \mathcal{M}$ that is homeomorphic to an open subset $\Omega \subset \mathbb{R}^d$. Such a homeomorphism $\phi : U \to \Omega$ is called a chart. (b): A tangent space can be defined at each point of the manifold. Its vectors are functionals $\mathcal{V}_{\gamma,\boldsymbol{p}}$. These can be represented numerically by associating the tangent space with a coordinate chart.

A manifold $\mathcal{M}$ is a d-dimensional topological space—a set of points with a certain structure (topology)—which has the appealing property that it is locally a real coordinate space $\mathbb{R}^d$. This allows points of the manifold to be represented numerically. The map that assigns numerical values to each point of the manifold is called a *(coordinate) chart*, i.e.

$$\phi : U \to \Omega, \tag{2.1}$$

maps a subset $U \in \mathcal{M}$ to a subset $\Omega \in \mathbb{R}^d$ (see also Figure 2.3a). Note that Ω cannot generally be associated with a vector space; elements of Ω are d-tuples and not coordinate vectors that are used for numerical linear algebraic computations. An *atlas* is a family of charts that covers the manifold.

TANGENT SPACE

At each point $\boldsymbol{p} \in \mathcal{M}$ one can define a vector space,

$$\mathcal{T}_{\boldsymbol{p}}\mathcal{M} = \{\mathcal{V}_{\gamma,\boldsymbol{p}} | \forall \gamma : \mathbb{R} \to \mathcal{M}\}, \tag{2.2}$$

consisting of functionals $\mathcal{V}_{\gamma,\boldsymbol{p}} : \mathcal{C}^\infty \to \mathbb{R}$. The functional $\mathcal{V}_{\gamma,\boldsymbol{p}} = (\cdots \circ \gamma)'[\gamma^{-1}[\boldsymbol{p}]]$[1] is a vector that is tangent to a path $\gamma(t)$ at $\boldsymbol{p}$.

The tangent vectors of all possible paths through $\boldsymbol{p}$ form the tangent space $\mathcal{T}_{\boldsymbol{p}}\mathcal{M}$. By associating the tangent space with a coordinate chart, a vector basis naturally arises (see also Figure 2.3b):

$$\begin{aligned}
\mathcal{V}_{\gamma,\boldsymbol{p}} &= (\cdots \circ \gamma)'[\gamma^{-1}[\boldsymbol{p}]], \\
&= (\cdots \circ (\phi^{-1} \circ \phi) \circ \gamma)'[\gamma^{-1}[\boldsymbol{p}]], && (2.3)\\
&= ((\cdots \circ \phi^{-1}) \circ (\phi \circ \gamma))'[\gamma^{-1}[\boldsymbol{p}]], && (2.4)\\
&= \underbrace{\partial_i (\cdots \circ \phi^{-1})[\phi \circ \gamma[t_{\boldsymbol{p}}]]}_{\frac{\partial \ldots}{\partial \phi^i}\big|_p} \underbrace{(\phi \circ \gamma)^{i\prime}[t_{\boldsymbol{p}}]}_{\omega^i}. && (2.5)
\end{aligned}$$

Here, we subsequently introduced the identity $\phi^{-1} \circ \phi$, re-associated, and applied the product rule in (2.3), (2.4) and (2.5), respectively. The sub and super indices i refer to Einstein's summation convention.

The basis vectors $\frac{\partial \ldots}{\partial \phi^i}\big|_p$ are the partial derivatives of the chart ϕ with respect to its d coordinates. This notation is usually written as $\frac{\partial}{\partial \phi^i}$, or ∂_i when the coordinate chart is clear from the context. Note that the empty spot in the partial derivative follows logically from the fact that basis vectors are elements of the tangent space: functionals.

The vector coordinates ω^i are real numbers as the time derivatives $(\phi \circ \gamma)^{i\prime} : \mathbb{R} \to \mathbb{R}$. They form the coordinate vectors

$$\mathfrak{w} = [\omega^0, \omega^1, \ldots, \omega^d]^\top, \tag{2.6}$$

that can be used to perform numerical vector algebra.

DISTANCE

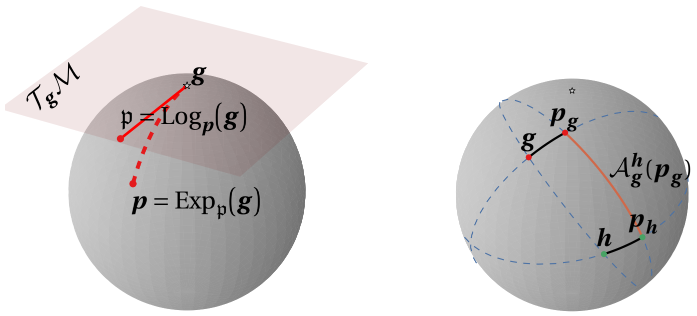
A manifold $\mathcal{M}$ with a Riemannian metric—a positive definite inner product $\langle \cdot, \cdot \rangle_{\boldsymbol{p}}$ defined on each tangent space $\mathcal{T}_{\boldsymbol{p}}\mathcal{M}$—is called a Riemannian manifold. The metric enables the length measurement of tangent vectors. As a result, it introduces the notion of length on the manifold. The length of a path $\gamma(t)$ between $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{M}$ is measured by

$$\mathcal{L}_{\boldsymbol{a}}^{\boldsymbol{b}}(\gamma) = \int_{\boldsymbol{a}}^{\boldsymbol{b}} \langle \dot{\gamma}(t), \dot{\gamma}(t) \rangle_{\gamma(t)} \, \mathrm{d}t. \tag{2.7}$$

Consequently, the distance between two points can be found by minimizing (2.7) for γ, i.e.

$$\operatorname{dist}(\boldsymbol{a}, \boldsymbol{b}) = \min \mathcal{L}_{\boldsymbol{a}}^{\boldsymbol{b}}(\gamma). \tag{2.8}$$

The path γ that minimizes (2.8) lies on a *geodesic*—the generalization of the straight line to Riemannian manifolds. The notion of minimum distance is essential in generalizing the Gaussian distribution, as it facilitates the definition of mean and dispersion (covariance).


$\mathcal{T}_{\boldsymbol{g}}\mathcal{M}$
$\boldsymbol{g}$
$\mathfrak{p} = \mathrm{Log}_{\boldsymbol{p}}(\boldsymbol{g})$
$\boldsymbol{p} = \mathrm{Exp}_{\mathfrak{p}}(\boldsymbol{g})$
$\boldsymbol{p_g}$
$\boldsymbol{g}$
$\mathcal{A}^{\boldsymbol{h}}_{\boldsymbol{g}}(\boldsymbol{p_g})$
$\boldsymbol{p_h}$
$\boldsymbol{h}$

(a) (b)

Figure 2.4: Manifold mappings and action function for $\mathcal{S}^2$. (a) The exponential and the logarithmic map provide local one-to-one mappings between the manifold and its tangent space at point $\boldsymbol{g}$. (b) An action $\mathcal{A}^{\boldsymbol{h}}_{\boldsymbol{g}}(\boldsymbol{p_g})$ maps $\boldsymbol{p_g}$ (a point defined relative to $\boldsymbol{g}$) to $\boldsymbol{p_h}$ by moving it along a geodesic (dotted lines) until it reaches a point such that the distance between $\boldsymbol{p_h}$ and $\boldsymbol{h}$ equals the distance between $\boldsymbol{p_g}$ and $\boldsymbol{g}$ (both distances visualized by black lines).

Exponential and Logarithmic Map

The tangent spaces and their bases provide the ability to perform linear algebra. In order to perform computations on the manifold, we need a distance preserving map to move points between the manifold and the tangent spaces. The exponential and logarithmic maps provide this functionality.

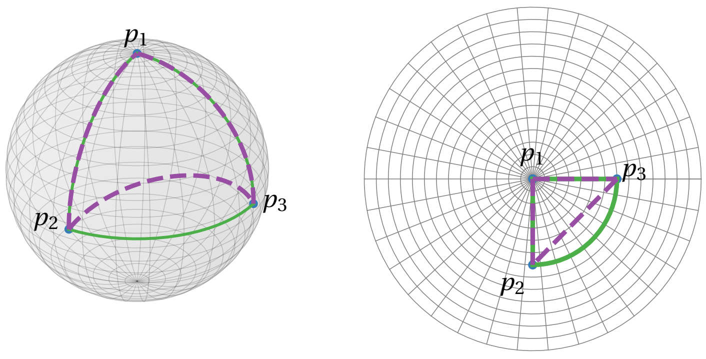
The *exponential map* $\mathrm{Exp}_{\boldsymbol{g}}(\cdot) : \mathcal{T}_{\boldsymbol{g}}\mathcal{M} \to \mathcal{M}$ is a distance preserving map from the tangent space to the manifold. $\mathrm{Exp}_{\boldsymbol{g}}(\mathfrak{p})$ maps $\mathfrak{p}$ to $\boldsymbol{p}$ in such a way that $\boldsymbol{p}$ lies on the *geodesic* through $\boldsymbol{g}$ with direction $\mathfrak{p}$, and the distance between $\boldsymbol{g}$ and $\boldsymbol{p}$ is $\|\mathfrak{p}\| = \langle \mathfrak{p}, \mathfrak{p} \rangle_{\boldsymbol{g}}$, see Figure 2.4a. The exponential only locally maps minimum distance. The set of all points for which the exponential does not map minimum distance is called the *cut-locus*. For spherical manifolds, which can be used to represent orientation with unit quaternions, the cut-locus only contains one point: the antipodal of the exponential base. For this point there is no single unique minimum distance path between the base and its antipodal.

The inverse of the exponential map is called the *logarithmic map* $\mathrm{Log}_{\boldsymbol{g}}(\cdot) : \mathcal{M} \to \mathcal{T}_{\boldsymbol{g}}\mathcal{M}$. The logarithmic map is defined for all points that do not lie in the cut-locus of its base, i.e. it is only defined when the exponential uniquely maps minimum distance paths.

The maps between the curved manifold and the linear tangent space need to straighten the curvature of the manifold. This straightening introduces deformations, as illustrated in Figure. 2.5. Figure 2.5b displays the tangent space of $\mathcal{S}^2$ at point $\boldsymbol{p}_1$. The circles and straight lines illustrate the latitudinal and longitudinal rings, respectively. Note that the latitudinal rings are stretched, while the longitudinal rings keep their original length. Consequently, the lengths of $\boldsymbol{p}_1 - \boldsymbol{p}_2$ and $\boldsymbol{p}_1 - \boldsymbol{p}_3$ reflect true distances as each pair lies on a longitudinal ring. The length $\boldsymbol{p}_2 - \boldsymbol{p}_3$ does not reflect the true distance, as the minimum

[1]This functional maps from the set of smooth functions $\mathcal{C}^\infty$ to the real numbers. It thus takes a function as argument. The ... indicate where the functional argument operates.

distance path between $\boldsymbol{p}_1$ and $\boldsymbol{p}_3$ appears curved, instead of straight in the projection (Figure 2.5b). Note that even the lengths of the circular paths, that appear in Figure 2.5b, do not reflect the true distances as they are stretched in the projection.



(a) Manifold $\mathcal{S}^2$ (b) Tangent space $\mathcal{T}_{\boldsymbol{p}_1}\mathcal{S}^2$

Figure 2.5: The manifold exponential and logarithmic maps deform the manifold-distances in order to make it fit in Euclidean space. Here, this deformation is illustrated for $\mathcal{S}^2$. The green lines illustrate the (projected) geodesics connecting $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ and $\boldsymbol{p}_3$. The purple lines illustrate the (projected) lines $\boldsymbol{p}_1 - \boldsymbol{p}_3$, $\boldsymbol{p}_1 - \boldsymbol{p}_2$ and $\boldsymbol{p}_2 - \boldsymbol{p}_3$.

In general, one exponential and logarithmic map is required for each tangent space. For homogeneous manifolds, however, their function can be moved from the origin $\boldsymbol{e}$ to other points on the manifold as follows

$$\mathrm{Exp}_{\boldsymbol{g}}(\mathfrak{p}_{\boldsymbol{g}}) = \mathcal{A}_{\boldsymbol{e}}^{\boldsymbol{g}}\left(\mathrm{Exp}_{\boldsymbol{e}}(\mathfrak{p}_{\boldsymbol{g}})\right), \tag{2.9}$$

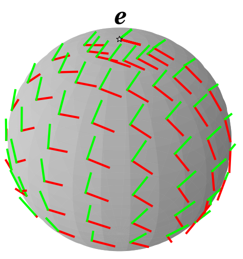
$$\mathrm{Log}_{\boldsymbol{g}}(\boldsymbol{p}) = \mathrm{Log}_{\boldsymbol{e}}\left(\mathcal{A}_{\boldsymbol{g}}^{\boldsymbol{e}}\left(\boldsymbol{p}\right)\right), \tag{2.10}$$

where $\mathcal{A}_{\boldsymbol{g}}^{\boldsymbol{h}}\left(\boldsymbol{p}_{\boldsymbol{g}}\right)$ is called the action function. It maps a point $\boldsymbol{p}_{\boldsymbol{g}}$ along a geodesic to $\boldsymbol{p}_{\boldsymbol{h}}$, in such a way that the distance between $\boldsymbol{p}_{\boldsymbol{g}}$ and $\boldsymbol{g}$ equals the distance between $\boldsymbol{p}_{\boldsymbol{h}}$ and $\boldsymbol{h}$ (see Figure 2.4b).

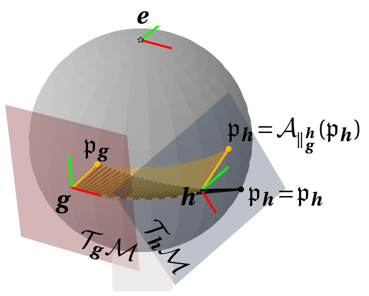
Action functions remove the need to compute a specific exponential and logarithmic map for each point in the manifold at the cost of imposing a specific alignment of the tangent bases. This is illustrated for $\mathcal{S}^2$ in Figure 2.6a. Although this alignment does not compromise the functions defined by (2.9) and (2.10), one must consider it while moving vectors from one tangent space to another.

Parallel Transport

Parallel transport moves vectors between two tangent spaces along the geodesic that connects the tangent bases; thereby maintaining a constant angle between the vector and the geodesic. To achieve parallel transport between any two points on the manifold, we need to compensate for the relative rotation between $\mathcal{T}_{\boldsymbol{g}}\mathcal{M}$ and $\mathcal{T}_{\boldsymbol{h}}\mathcal{M}$. For d-spheres



(a)


e
$\mathfrak{p}_g$
$\mathfrak{p}_h = \mathcal{A}_{\|_g^h}(\mathfrak{p}_h)$
g
h
$\mathfrak{p}_h = \mathfrak{p}_h$
$\mathcal{T}_g\mathcal{M}$
$\mathcal{T}_h\mathcal{M}$

(b)

Figure 2.6: (a) Tangent space alignment with respect to $\boldsymbol{e}$. (b) Even though the tangent bases are aligned with respect to $\boldsymbol{e}$, base misalignment exists between $\mathcal{T}_{\boldsymbol{g}}\mathcal{M}$ and $\mathcal{T}_{\boldsymbol{h}}\mathcal{M}$ when one of them does not lie on a geodesic through $\boldsymbol{e}$. In such cases, parallel transport of $\mathfrak{p}$ from $\mathcal{T}_{\boldsymbol{g}}\mathcal{M}$ to $\mathcal{T}_{\boldsymbol{h}}\mathcal{M}$ requires a rotation $\mathcal{A}_{\|_{\boldsymbol{g}}^{\boldsymbol{h}}}(\mathfrak{p})$ to compensate for the misalignment of the tangent spaces.

this rotation is given by

$$\boldsymbol{R}_{\|_{\boldsymbol{g}}^{\boldsymbol{h}}} = \boldsymbol{I}_{d+1} - \sin(m)\,\boldsymbol{g}\boldsymbol{u}^{\top} + (\cos(m)-1)\,\boldsymbol{u}\boldsymbol{u}^{\top}, \tag{2.11}$$

where $\boldsymbol{u} = [\mathfrak{v}^{\top}, 0]^{\top}$ gives the direction of transportation. It is constructed from $\boldsymbol{h}$ by mapping it into $\mathcal{T}_{\boldsymbol{g}}\mathcal{M}$, normalizing it, and finally rotating it to $\boldsymbol{g}$; i.e. $\mathfrak{v} = \boldsymbol{R}_{\boldsymbol{e}}^{\boldsymbol{g}}\,\mathrm{Log}_{\boldsymbol{g}}(\boldsymbol{h})\,/\,m$ with $m = ||\,\mathrm{Log}_{\boldsymbol{g}}(\boldsymbol{h})\,||$ the angle of transportation (See [67], Ch. 8). Notice that (2.11) is defined in the manifold ambient space $\mathbb{R}^{d+1}$, while we have defined our tangent spaces in $\mathbb{R}^{d}$. To achieve parallel transport between $\mathcal{T}_{\boldsymbol{g}}\mathcal{M}$ and $\mathcal{T}_{\boldsymbol{h}}\mathcal{M}$, we define the parallel action

$$\mathcal{A}_{\|_{\boldsymbol{g}}^{\boldsymbol{h}}}(\mathfrak{p}) = \boldsymbol{B}^{\top}\,\boldsymbol{R}_{\boldsymbol{h}}^{\boldsymbol{e}}\,\boldsymbol{R}_{\|_{\boldsymbol{g}}^{\boldsymbol{h}}}\,\boldsymbol{R}_{\boldsymbol{e}}^{\boldsymbol{g}}\,\boldsymbol{B}\,\mathfrak{p}, \tag{2.12}$$

where $\boldsymbol{B}$ contains the direction of the bases at the origin. For the manifolds $\mathcal{S}^2$ and $\mathcal{S}^3$ we use

$$\boldsymbol{B}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}^{\top}, \text{ and } \boldsymbol{B}_3 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{\top}. \tag{2.13}$$

Furthermore, $\boldsymbol{R}_{\boldsymbol{e}}^{\boldsymbol{g}}$ and $\boldsymbol{R}_{\boldsymbol{h}}^{\boldsymbol{e}}$ represent rotations between $\boldsymbol{e}$ and $\boldsymbol{g}$, and $\boldsymbol{h}$ and $\boldsymbol{e}$, respectively. Note that no information is lost through the projection $\boldsymbol{B}$, which can be understood by realizing that the parallel action is invertible. The effect of parallel transport is visualized for $\mathcal{S}^2$ in Figure 2.6b.

Cartesian Product

Finally, we note that the Cartesian product of two Riemannian manifolds is again a Riemannian manifold. This property allows us to define joint distributions on any combination of Riemannian manifolds. For example, a robot pose is represented by the Cartesian product of a 3 dimensional Euclidean space and a hypersphere, i.e. $\boldsymbol{p} \in \mathbb{R}^3 \times \mathcal{S}^3$.

The corresponding Exp(), Log(), $\mathcal{A}$ (), and parallel transport of the Cartesian product are obtained by concatenating the individual functions, e.g.

$$\mathrm{Log}_{\begin{bmatrix}\boldsymbol{e_a}\\ \boldsymbol{e_b}\end{bmatrix}}\left(\begin{bmatrix}\boldsymbol{a}\\ \boldsymbol{b}\end{bmatrix}\right) = \begin{bmatrix}\mathrm{Log}_{\boldsymbol{e_a}}(\boldsymbol{a})\\ \mathrm{Log}_{\boldsymbol{e_b}}(\boldsymbol{b})\end{bmatrix}.$$

$\mathcal{M}$:	$\mathbb{R}^d$	$\mathcal{S}^2$	$\mathcal{S}^3$
$\boldsymbol{g} \in \mathcal{M}$	$\begin{pmatrix} g_{x_1} \\ \vdots \\ g_{x_d} \end{pmatrix}$	$\begin{pmatrix} g_x \\ g_y \\ g_z \end{pmatrix}$	$\begin{pmatrix} q0 \\ \boldsymbol{q} \end{pmatrix}$
$\mathrm{Exp}_{\boldsymbol{e}}(\mathfrak{g})$	$\boldsymbol{e}+\begin{bmatrix} \mathfrak{g}_{x_1} \\ \vdots \\ \mathfrak{g}_{x_d} \end{bmatrix}$	$\begin{cases} \begin{pmatrix} \mathrm{s}(\Vert\mathfrak{g}\Vert)\frac{\mathfrak{g}}{\Vert\mathfrak{g}\Vert} \\ \mathrm{c}(\Vert\mathfrak{g}\Vert) \end{pmatrix}, & \Vert\mathfrak{g}\Vert\neq 0 \\ \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, & \Vert\mathfrak{g}\Vert = 0 \end{cases}$	$\begin{cases} \begin{pmatrix} \mathrm{c}(\Vert\mathfrak{g}\Vert) \\ \mathrm{s}(\Vert\mathfrak{g}\Vert)\frac{\mathfrak{g}}{\Vert\mathfrak{g}\Vert} \end{pmatrix}, & \mathfrak{g}\Vert\neq 0 \\ \begin{pmatrix} 1 \\ \boldsymbol{0} \end{pmatrix}, & \Vert\mathfrak{g}\Vert = 0 \end{cases}$
$\mathrm{Log}_{\boldsymbol{e}}(\boldsymbol{g})$	$\begin{bmatrix} g_{x_1} \\ \vdots \\ g_{x_d} \end{bmatrix}$	$\begin{cases} \mathrm{ac}(g_z)\frac{[g_x,g_y]^\top}{\Vert[g_x,g_y]^\top\Vert}, & g_z\neq 1 \\ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, & g_z = 1 \end{cases}$	$\begin{cases} \mathrm{ac}_*(q_0)\frac{\boldsymbol{q}}{\Vert\boldsymbol{q}\Vert}, & q_0\neq 1 \\ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, & q_0 = 1 \end{cases}$
$\mathcal{A}_{\boldsymbol{g}}^{\boldsymbol{h}}(\boldsymbol{p})$	$\boldsymbol{p}_g - \boldsymbol{g} + \boldsymbol{h}$	$\boldsymbol{R}_{\boldsymbol{e}}^{\boldsymbol{h}}\boldsymbol{R}_{\boldsymbol{g}}^{\boldsymbol{e}}\boldsymbol{p}_{\boldsymbol{g}}$	$\boldsymbol{h} * \boldsymbol{g}^{-1} * \boldsymbol{p}_{\boldsymbol{g}}$
$\mathcal{A}_{\Vert\boldsymbol{g}}^{\boldsymbol{h}}(\mathfrak{p})$	$\boldsymbol{I}_d\mathfrak{p}$	$\boldsymbol{B}_2^\top\boldsymbol{R}_{\boldsymbol{h}}^{\boldsymbol{e}}\boldsymbol{R}_{\Vert\boldsymbol{g}}^{\boldsymbol{h}}\boldsymbol{R}_{\boldsymbol{e}}^{\boldsymbol{g}}\boldsymbol{B}_2\mathfrak{p}$	$\boldsymbol{B}_3^\top\boldsymbol{Q}_{\boldsymbol{h}}^\top\boldsymbol{R}_{\Vert\boldsymbol{g}}^{\boldsymbol{h}}\boldsymbol{Q}_{\boldsymbol{g}}\boldsymbol{B}_3\mathfrak{p}$

Table 2.1: Overview of the exponential and logarithmic maps, and the action and parallel transport functions for all Riemannian manifolds considered in this work. $\mathrm{s}(\cdot), \mathrm{c}(\cdot), \mathrm{ac}_*(\cdot)$ are short notations for the sine, cosine, and a modified version of the arccosine[2], respectively. The elements of $\mathcal{S}^3$ are quaternions, $*$ defines their product, and $^{-1}$ a quaternion inverse. $\boldsymbol{Q}_{\boldsymbol{g}}$ and $\boldsymbol{Q}_{\boldsymbol{h}}$ represent the quaternion matrices of $\boldsymbol{g}$ and $\boldsymbol{h}$.

2.2.2. ORIENTATION REPRESENTATION

The relative orientation between two coordinate systems is defined by a linear operator, a $d \times d$ orthonormal matrix with determinant 1. The set of all rotation matrices forms a group under the matrix product. This group is called the special orthogonal group,

$$SO(n) = \{R \in \mathbb{R}^{d\times d} | RR^T = R^T R = I, \det(R) = 1\}. \tag{2.14}$$

Although a rotation matrix has 9 numerical entries, a change in orientation is characterized by only 3 degrees of freedom. Yet, there exists no 3-parameter representation of $SO(3)$ that is singularity free, and uniquely covering [68]. Singularity free, refers to the existence of a global continuous function $f : SO(3) \to \mathbb{R}^3$.

[2]The space of unit quaternions, $\mathcal{S}^3$, provides a double covering over rotations. To ensure that the distance between two antipodal rotations is zero we define $\mathrm{arccos}_*(\rho)\begin{cases}\mathrm{arccos}(\rho) - \pi & ,-1 \le \rho < 0 \\ \mathrm{arccos}(\rho) & ,0 \le \rho \le 1\end{cases}$.

As $SO(3)$ is not Euclidean, standard statistical methods will not respect its manifold structure—the Euclidean mean of a set of rotation matrices is not a rotation matrix. Local 3-parameter representations of $SO(3)$ appear to be more forgiving in this regard. In this section, we will see that this appearance can be deceiving. We review common three and four parameter representations of orientation, namely: Euler angles, axis-angle and unit quaternions. We discuss their relation to $SO(3)$ and their ability to compose orientation through summation. To aid this discussion, we first introduce additional details of $SO(3)$. The description of the different representations is based on a more elaborate discussion given by Murray *et al.* [68] (Ch. 2.2, and Appx. A).

SPECIAL ORTHOGONAL GROUP $SO(3)$

$SO(3)$ is both a compact matrix Lie-group and a Riemannian manifold under the common Euclidean metric. Each Lie-group has a Lie-algebra, a tangent space defined at the identity. The elements of the Lie-algebra of $SO(3)$, denoted $\mathfrak{so}(3)$, are 3×3 skew-symmetric matrices

$$\tilde{\boldsymbol{\omega}} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{2.15}$$

The set of such matrices is a $3D$ Euclidean manifold, whose elements can be represented as $\boldsymbol{\omega} \in \mathbb{R}^3$. $\boldsymbol{\omega}$ is often described as the angular velocity required to move from the identity to a particular orientation in unit time.

The matrix exponential provides a mapping between the matrix Lie-group and its algebra. In the special case of $SO(3)$, it can efficiently be computed using Rodriguez' formula for rotation

$$\boldsymbol{R}(\boldsymbol{\omega}) = \boldsymbol{I} + \sin(\theta)\tilde{\bar{\boldsymbol{\omega}}} + (1-\cos(\theta))\tilde{\bar{\boldsymbol{\omega}}}^2, \tag{2.16}$$

where $\boldsymbol{\omega} = \theta\bar{\boldsymbol{\omega}}$ is composed of an angle $\theta = |\boldsymbol{\omega}|$ and a unit vector $\bar{\boldsymbol{\omega}} = \boldsymbol{\omega}/\theta$. The operator $\tilde{\cdot}$, transforms $\boldsymbol{\omega}$ in its skew-symmetric representation. Because the exponential map relates $\boldsymbol{\omega}$ to $\boldsymbol{R}$, $\boldsymbol{\omega}$ is sometimes called the exponential coordinate of $\boldsymbol{R}$. The inverse of operation is achieved using the logarithmic map. Inverting Rodriguez' formula, it can be efficiently computed, namely

$$\theta = \begin{cases} \arccos\left(\frac{1-\mathrm{trace}(\boldsymbol{R})}{2}\right) & \text{if } \boldsymbol{R} \neq \boldsymbol{I}, \\ 2\pi k & \text{otherwise,} \end{cases} \tag{2.17}$$

$$\bar{\boldsymbol{\omega}} = \begin{cases} \frac{1}{\sin(\theta)}\left(\boldsymbol{R} - \boldsymbol{R}^\top\right) & \text{if } \boldsymbol{R} \neq \boldsymbol{I}, \\ \boldsymbol{0} & \text{otherwise .} \end{cases} \tag{2.18}$$

Here, the integer k and $\bar{\boldsymbol{\omega}}$ can be arbitrary chosen when $\boldsymbol{R} = \boldsymbol{I}$. Note that the inverse of the exponential map is not continuous near the identity. The exponential map is a many-to-one map: if $\bar{\boldsymbol{\omega}}\theta$ relates to $\boldsymbol{R}$, so will $\bar{\boldsymbol{\omega}}(\theta+2\pi k)$. Representations of orientation that rely on $\boldsymbol{\omega}$ are therefore not uniquely covering $SO(3)$. Note that the Lie-group exponential and logarithmic maps for $SO(3)$ coincide with the Riemannian exponential and logarithmic maps.

EULER ANGLES

Euler angles describe a rotation $\boldsymbol{R} \in SO(3)$ using three scalars. Each scalar describes a rotation about a (local) coordinate axis. By concatenating the individual rotations one obtains the desired rotation matrix, e.g.

$$\boldsymbol{R}_{xyz}(\psi,\phi,\theta) = \boldsymbol{R}_x(\psi)\boldsymbol{R}_y(\phi)\boldsymbol{R}_z(\theta), \tag{2.19}$$

describes the rotation by subsequently rotating data about the local z, y and x axes. Here, the order and axes of rotation in the sequence may change depending on the application. Intuitively, Euler angles can be visualized as a stack of rotations [69].

Euler angles are not coordinate vectors—they do not correspond to a single coordinate basis—instead they form tuple. To illustrate this, we rewrite (2.19) by replacing the rotations $\boldsymbol{R}_x$, $\boldsymbol{R}_y$, $\boldsymbol{R}_z$, by their exponential maps

$$\begin{aligned}\boldsymbol{R}_{xyz}(\psi,\phi,\theta) &= \operatorname{Exp}_{\boldsymbol{I}}\big(\boldsymbol{e}_x\psi\big)\ \operatorname{Exp}_{\boldsymbol{I}}\big(\boldsymbol{e}_y\phi\big)\ \operatorname{Exp}_{\boldsymbol{I}}\big(\boldsymbol{e}_z\theta\big) &(2.20)\\ &\neq \operatorname{Exp}_{\boldsymbol{I}}\big(\boldsymbol{e}_x\psi + \boldsymbol{e}_y\phi + \boldsymbol{e}_z\theta\big) &(2.21)\end{aligned}$$

where $\{\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z\}$ are basis vectors. Note that we cannot generally apply

$$\operatorname{Exp}_{\boldsymbol{I}}(\boldsymbol{A})\operatorname{Exp}_{\boldsymbol{I}}(\boldsymbol{B}) = \operatorname{Exp}_{\boldsymbol{I}}(\boldsymbol{A}+\boldsymbol{B}), \tag{2.22}$$

as this property requires the condition $\boldsymbol{AB} = \boldsymbol{BA}$ [70]. Consequently, we conclude that Euler angles are not coordinate vectors because they cannot be expressed in a single vector space. Instead, they can be seen as a 3-tuple of a coordinate chart that covers $SO(3)$.

Although summation of Euler angles will yield valid rotation matrices, this summation does not properly compose orientation, since summation is not performed in a single vector space. Similarly, one might naively compute a correlation matrix over Euler angles. The resulting covariance matrix, however, will not properly describe the correlation among the different rotational degrees of freedom. In fact, the resulting matrix is not a proper covariance matrix because it is not a tensor—A mapping which takes two elements form the same vector space to the real numbers, $\mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$.

AXIS-ANGLE

Euler stated that any orientation $\boldsymbol{R} \in SO(3)$ is equivalent to a rotation about a fixed axis $\boldsymbol{\omega} \in \mathbb{R}^3$ through an angle $\theta \in [0, 2\pi)$. This description of rotation is often denoted the *axis-angle* or *equivalent axis* representation. When the axis of rotation has unit norm, i.e. $|\omega| = 1$, the axis-angle representation corresponds to the exponential coordinates of $SO(3)$.

The axis-angle representation allows the representation of orientation in a vector space (the Lie-algebra). Unlike Euler angles, the axis-angle representation thus formally allows Euclidean operations. However, the difference between two orientations does not correspond to their difference in axis-angle representation, i.e.

$$\operatorname{Log}_{\boldsymbol{I}}\big(\boldsymbol{R}_1\boldsymbol{R}_2^{\top}\big) \neq \operatorname{Log}_{\boldsymbol{I}}(\boldsymbol{R}_1) - \operatorname{Log}_{\boldsymbol{I}}(\boldsymbol{R}_2)\,,\ \text{if } \boldsymbol{R}_1 \neq \boldsymbol{I}, \boldsymbol{R}_2 \neq \boldsymbol{I}. \tag{2.23}$$

This distortion results from ‘flattening’ $SO(3)$ onto $\mathbb{R}^3$ through the exponential (see also Section 2.2.1).

UNIT QUATERNION

A quaternion is a tuple consisting of four scalars often grouped as a scalar q_0 and a $3D$ vector $\boldsymbol{q}_v$ [71]. The set of unit quaternions (quaternions with norm 1), covers the rotational group $SO(3)$ twice: each element $\boldsymbol{R} \in SO(3)$ corresponds to two unit quaternions, namely $\boldsymbol{q}$ and $-\boldsymbol{q}$. Therefore, the unit quaternions do not provide a unique representation of $SO(3)$. However, the maps from $SO(3) \to \mathcal{S}^3$ and $\mathcal{S}^3 \to SO(3)$ are both continuous. The unit quaternions thus provide singularity free representation of $SO(3)$.

Composition of orientation, is achieved through the quaternion product. As $\mathcal{S}^3$ is not a Euclidean space, unit quaternions cannot be composed under summation. Similarly to $SO(3)$, $\mathcal{S}^3$ is a Riemannian manifold. Distances between unit quaternions are thus computable in an appropriate tangent space.

The elements of its tangent spaces have a direct relation with the velocity vectors of $\mathfrak{so}(3)$, namely:

$$\boldsymbol{\beta} = \frac{1}{2}\boldsymbol{\omega}, \tag{2.24}$$

where $\boldsymbol{\beta} \in \mathcal{T}_{\boldsymbol{e}}\mathcal{S}^3$ is an element of the tangent space at the identity $\boldsymbol{e}$ of $\mathcal{S}^3$, and $\boldsymbol{\omega} \in \mathfrak{so}(3)$ the corresponding velocity vector in $SO(3)$. The intuition behind the scaling factor $\frac{1}{2}$ is the double covering of rotation: the equator length of $\mathcal{S}^3$ is 2π, which corresponds to $2 \cdot 2\pi$ rotation.

COMPARISON

We assess suitability of the presented parameterizations using Table 2.2. The first three criteria appear commonly in comparisons of orientation parameterizations: having a minimal (3-parameter) representation can be desirable because such parameterizations can be considered 'unconstrained'. Singularity free representations are desirable in applications with large orientation range. Finally, uniqueness ensures an unambiguous mapping between $SO(3)$ and its parameterization. The remaining criteria are less common, yet essential for statistical applications. First, we require the parameterization to form a vector space to enable computation of mean and covariance[3]. Furthermore, the parameterization needs to measure the true minimum distance between rotations.

Table 2.2 separately lists $\mathcal{T}_{\boldsymbol{R}}SO(3)$ and $\mathcal{T}_{\boldsymbol{q}}\mathcal{S}^3$. The explicit use of tangent spaces enables measurement of distance, an essential element for our statistical application. However, both groups can only measure minimum distance between two orientations $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ when they lie on a geodesic that crosses the origin of the tangent space. In any other case, the deformation introduced by the logarithmic map alters the measured length (see also Section 2.2.1). Given the non-linear nature of $SO(3)$ such a limitation is inevitable. The fact that we can measure minimum distance by taking into account these limitations, opens a way for statistics on the manifold.

$\mathcal{T}_{\boldsymbol{R}}SO(3)$ and $\mathcal{T}_{\boldsymbol{q}}\mathcal{S}^3$ are neither minimal nor unique. They are not considered minimal, as a rotation expressed in a tangent space requires both the tangent vector and the tangent base (i.e. $\boldsymbol{q} \in \mathcal{S}^3$ or $\boldsymbol{R} \in SO(3)$). Their parameterizations are not unique because

[3]In Section 2.5.1 we will use mean-point that is not defined in a vector space. But its computation—and that of the covariance—still requires a vector space.

their exponential maps are many-to-one (as discussed before multiple points in the tangent space correspond to the same rotation). This is not problematic in our application as long as the logarithmic maps project minimum distance. This is the case for the logarithmic map of $SO(3)$ (2.18), and can be achieved for the unit quaternions. Recall that the group of unit quaternions covers $SO(3)$ twice, the measurement of distance between two orientations $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ is thus ambiguous. This is resolved by defining the logarithmic map of $\mathcal{S}^3$ in such a way that $\boldsymbol{q}$ and $-\boldsymbol{q}$ are mapped to the same point in $\mathcal{T}_{\boldsymbol{q}}\mathcal{S}^3$ (see Table 2.1). This creates a unique map from $SO(3)$ to $\mathcal{T}_{\boldsymbol{q}}\mathcal{S}^3$. Using uniquely mapping logarithmic maps, both mean and covariance can be computed.

The map from $SO(3)$ to $\mathcal{T}_{\boldsymbol{q}}\mathcal{S}^3$ is singularity free when using the matrix exponential and logarithmic maps. A practical implementation requires the use of Rodriguez' formula, which is undefined around the origin of the tangent space. Practically, this forms no restriction as we can sufficiently approximate these maps around origin using the first two terms of their Taylor expansion.

Concluding, both $\mathcal{T}_{\boldsymbol{q}}\mathcal{S}^3$ and $\mathcal{T}_{\boldsymbol{R}}SO(3)$ provide a suitable orientation parameterization for use in statistics. In Section 2.5.1, it turns out that the use of unit quaternions through $\mathcal{T}_{\boldsymbol{q}}\mathcal{S}^3$ can be advantageous because its double covering of orientation makes distributions on orientation more dense.

Parameterization	minimal	sf	unique	vector space	min. dist.
$SO(3)$	–	✓	✓	–	–
$\mathcal{T}_{\boldsymbol{R}}SO(3)$	–	✓*	–	✓	✓*
Axis-angle	✓	–	–	✓	–
Euler angles	✓	–	–	–	–
Unit quaternion	–	✓	–	–	–
$\mathcal{T}_{\boldsymbol{q}}\mathcal{S}^3$	–	✓*	–	✓	✓*

Table 2.2: Comparison of parameterizations of $SO(3)$ using the criteria (left-to-right): Minimal, referring to the ability to represent orientation by the minimum number of parameters; sf (Singularity-Free), indicates that the mappings between a parameterization and $SO(3)$ are continuous and always existing; unique, indicates that each element of $SO(3)$ is uniquely defined in the parameterization; vector space, indicates the parameterization is a vector space; min. dist. (minimum distance), indicates that the minimum distance between orientations can be properly measured. An asterisk indicates that the condition holds under specific conditions (see comparison in Section 2.2.2).

2.3. Related Work

This thesis is originally motivated by the desire to perform statistics on a combination of position and orientation data. In this section we review methods used in robotics and PbD which apply statistics on a combination of position orientation.

Orientation statistics are related to the field of directional statistics [72], a study that is mainly concerned with observations in the form of unit vectors. Mardia *et al.* [72] distinguish three approaches within this field:

- *Ambient statistics*, which neglect the structure of the underlying manifold and treat the space as being Euclidean;

- *Wrapped (manifold) statistics*, which rely on a statistical distribution that is wrapped—tailored—on the manifold;

- *Intrinsic statistics*, which perform statistics indirectly on the manifold using a (local) bijective mapping between the directional manifold and other (Euclidean) manifolds.

Although parameterizations for orientations are not limited to unit vector data, this categorization is well-suited to group the related work.

AMBIENT APPROACHES

Within the field of PbD, various examples of the ambient approach can be found. Calinon *et al.* [18] use Euler angles to encode simple orientation (alignment) in a Task-Parameterized Gaussian Mixture Model (TP-GMM). As pointed out by Silvério *et al.* [41] the approach cannot be used for general orientations. Instead, Silvério *et al.* propose to use the quaternion representation for orientation as this allows linear composition of orientation through the quaternion matrix. However, the proposed method still assumes the quaternion data to be embedded in a Euclidean space, thereby neglecting its unit constraint. As a result, additional regularization steps are required in both the training and reproduction phase. Furthermore, the extracted covariance matrix is 4 dimensional, while orientation has 3 degrees of freedom. Malekzadeh *et al.* [73] used this approach to program end-effector pose to a bionic handling assistant.

To avoid the unit constraint of the quaternion, Gribovskaya *et al.* [74] rely on the axis-angle representation. They demonstrate how orientation can be incorporated in a dynamical system. The dynamical system is represented using a GMM, encoding the joint distribution over axis $\boldsymbol{s}$, angle θ and rotational velocity $\boldsymbol{\omega}$: $\mathcal{P}(\boldsymbol{\omega},\theta,\boldsymbol{s})$. The dynamical system is then obtained by computing the conditional distribution $\mathcal{P}(\boldsymbol{\omega}|\theta,\boldsymbol{s})$ using Gaussian Mixture Regression (GMR).

Kim *et al.* [75] proposed to encode orientation using the first two columns of the rotation matrix. As indicated later by Kim *et al.* [42], this approach requires additional post-processing to meet orthonormality constraints, and does not pernit a proper computation of probability because of an incorrect distance measure between rotations.

WRAPPED APPROACHES

The examples of wrapped approaches to encode and synthesize orientation data are scarce in Robotics. There are, however, examples of wrapped approaches for pose estimation. In [76–79], the Bingham distribution is used for orientation estimation and filtering. The Bingham distribution is an anti-podal symmetric distribution on a unit hypersphere. The anti-podal aspect of the distribution makes it especially suited to encode unit quaternions, as anti-podal quaternions represent the same orientation.

INTRINSIC APPROACHES

Feiten, Lang *et al.* [59, 60] perform statistics on, respectively, orientation and pose using the central projection method. The orientation and pose data, represented as (dual) quaternions, is projected in a Euclidean space using the central projection. Within this Euclidean space, distributions are computed on the projected data: Feiten *et al.* [60]

approximate the distribution using a (mixture of) Gaussian(s), while Lang *et al.* [59] use a Gaussian Process (GP). Since the central projection is not distance preserving (i.e. points equally spaced on the manifold will not be equally spaced in the projection plane), the projection plane cannot be directly used to compute statistics on manifold elements. More recently, Lang *et al.* modified their approach based on GP by replacing the central projection method with a distance preserving measure [38].

Ude *et al.* [26] propose methods to include orientation expressed as rotation matrices or quaternions in the Dynamic Movement Primitives (DMP) framework [8]. As the forcing term of DMP is a Euclidean construct, its definition required projection of the orientation error—the difference between the current and the goal orientation—into a tangent space of the rotational manifold.

Both Simo-Serra *et al.* [51] and Kim *et al.* [42] follow a Riemannian approach (although not explicitly mentioned in [42]). Simo-Serra *et al.* use their method for human pose tracking, and Kim *et al.* apply their method to motion modeling and synthesis. Both methods consider rotational data as elements of the unit quaternion manifold, and encode a distribution on orientation using a Riemannian Gaussian [47]. Independently, both works present a method for Gaussian conditioning. In this thesis, we follow a similar approach, but propose an alternative method for Gaussian conditioning. The newly proposed generalization of Gaussian conditioning follows the manifold geodesics, yielding a proper generalization of Gaussian conditioning (see Section 2.5.4).

The need to express uncertainty over orientation or pose also arises in state propagation problems such as Kalman filtering or SLAM [53, 56, 80–83]. These methods seem to have been developed independently of [42, 47, 51]. Noteworthy is the work of Barfoot *et al.* [56] who study distributions of uncertainty on the Lie-groups $SO(3)$ and $SE(3)$. They derive these distribution starting from a standard Euclidean Gaussian. Like in Pennec *et al.* [47], they recognized that the normalization in the resulting Gaussian-like expression cannot be computed easily. However, they show that explicit computation of this expression is not required for their application.

2.4. PARAMETERIZATION OF END-EFFECTOR POSE

End-effector pose can be represented as elements of $\mathrm{SE}(3)$, $\mathrm{SO}(3)\times\mathbb{R}^3$ or $\mathcal{S}^3\times\mathbb{R}^3$. In this thesis, we represent end-effector pose as elements of $\mathcal{S}^3\times\mathbb{R}^3$: the Cartesian product of unit quaternions and 3D translations. This section describes the distinction between $\mathrm{SE}(3)$ and $\mathrm{SO}(3)\times\mathbb{R}^3$ and motivates our choice for selecting the Cartesian product of rotation and translation over $\mathrm{SE}(3)$. The argument uses the rotational group $\mathrm{SO}(3)$ to maintain consistency with the cited work; but, the same argument holds when the rotational group is represented by unit quaternions. Our preference for unit quaternions over $\mathrm{SO}(3)$ is motivated in Section 2.5.1.

The pose of an end-effector is described by a position and orientation in space, which are quantified by elements of $\mathbb{R}^3$ and $\mathrm{SO}(3)$, respectively. Different topologies can be assigned to the set of poses. For example, both $\mathrm{SO}(3)\times\mathbb{R}^3$ and $\mathrm{SE}(3)$ contain the same set of poses under a different topology. The former is the Cartesian product of the rotational and translational group, and the latter the group of rigid-body transformations, i.e.

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{v} \\ \boldsymbol{0} & 1 \end{bmatrix} \in \mathrm{SE}(3). \tag{2.25}$$

These transformations map Euclidean vectors between vector spaces while preserving their relative distance—as if they belong to a rigid-body.

SE(3) and $\mathrm{SO}(3) \times \mathbb{R}^3$ are both groups and manifolds. Yet, $\mathrm{SE}(3) \neq \mathrm{SO}(3) \times \mathbb{R}^3$, as their topology and group structure differ. The difference between SE(3) and $\mathrm{SO}(3) \times \mathbb{R}^3$ becomes apparent in their group operation $*$. Consider the composition of two poses $(\boldsymbol{A}, \boldsymbol{b})$ and $(\boldsymbol{R}, \boldsymbol{v})$. For $\mathrm{SO}(3) \times \mathbb{R}^3$ this yields:

$$\begin{pmatrix} \boldsymbol{AR} \\ \boldsymbol{b} + \boldsymbol{v} \end{pmatrix} = \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{b} \end{pmatrix} * \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{v} \end{pmatrix}, \tag{2.26}$$

and for SE(3) this yields:

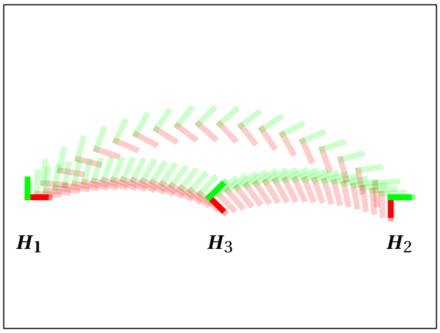
$$\begin{bmatrix} \boldsymbol{AR} & \boldsymbol{Av} + \boldsymbol{b} \\ \boldsymbol{0} & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{b} \\ \boldsymbol{0} & 1 \end{bmatrix} * \begin{bmatrix} \boldsymbol{R} & \boldsymbol{v} \\ \boldsymbol{0} & 1 \end{bmatrix}. \tag{2.27}$$

Clearly, the translational part is transformed differently, as $\boldsymbol{Av} + \boldsymbol{b} \neq \boldsymbol{v} + \boldsymbol{b}$. The rigid-body structure is not preserved in the group operation of $\mathrm{SO}(3) \times \mathbb{R}^3$, thereby causing the inequality $\boldsymbol{Av} + \boldsymbol{b} \neq \boldsymbol{v} + \boldsymbol{b}$.

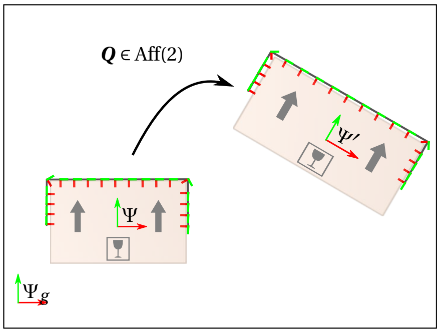
Rigid-body transformations can be viewed as rigid-body displacement in a common inertial frame. A well-known result from kinematics states: "Any rigid-body displacement can be realized by a rotation about an axis combined with a translation parallel to that axis"(Chasles' theorem) [68, 84]. In other words, any rigid-body displacement can be described by screw motion. Screw motions provide a coordinate independent description of rigid-body velocity—they are independent of inertial frame. This is a very convenient property, which enables coordinate-free descriptions of rigid-body dynamics, and efficient algorithms to solve them [85, 86].

To determine if the structure of SE(3) suits our needs, we ask the following questions: Are rigid-body displacements parameterized by screw motions minimum distance paths; i.e. are they geodesics? Do we require a coordinate independent representation? Does our application require rigid-body transformations?

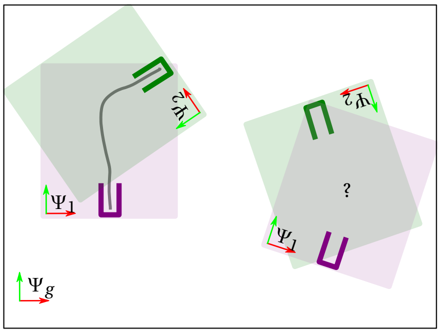
Briefly answered: screw motions do not generally represent minimum distance [84, 87]; our approach requires a coordinate-dependent approach, and transformations of end-effector pose that are not restricted to rigid-body transformations (see Chapter 4). $\mathrm{SO}(3) \times \mathbb{R}^3$, on the other hand, does provide the means to measure minimum distance and transform rotation and orientation separately. We motivate these answers below using the illustrative examples in Figure 2.7.



(a) The screw motion in SE(2) between two poses does not generally lie on a minimal distance path. The opaque red/green lines indicate poses, the transparent pose-sequences depict screw motion paths between poses.



(b) An affine transformation applied to a 2D box. It changes the box scaling, translation and rotation. In order to transform a pose-path, the full affine transformation is applied to the spatial part, but only the rotational element is applied to the orientation part.



(c) A conceptual illustration of the context-adaptive approach that is introduced in Chapter 4. The approach models tasks in multiple local coordinate systems that relate to objects in the environment. By moving the local coordinate systems with the object location context adaptation is achieved.

Figure 2.7: Illustrations used in Section 2.4 to motivate the parameterization of end-effector pose by $SO(3) \times \mathbb{R}^3$.

METRICS, GEODESICS AND SCREW MOTIONS ON SE(3)

It is well-known that there exists no bi-invariant[4], or a natural left or right invariant metric on SE(3) [68, 84, 87, 88]. Park and Brockett [87, 89] showed that

$$\boldsymbol{G} = \begin{bmatrix} \alpha \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \beta \boldsymbol{I} \end{bmatrix}, \tag{2.28}$$

[4] Bi-invariance implies that the metric holds under both left and right group operations. The distance $\mathrm{dist}(\cdot,\cdot)_{\boldsymbol{G}}$ on manifold $\mathcal{M}$ measured under metric $\boldsymbol{G}$ is bi-invariant if $\mathrm{dist}(\boldsymbol{a},\boldsymbol{b})_{\boldsymbol{G}} = \mathrm{dist}(\boldsymbol{ebe},\boldsymbol{eae})_{\boldsymbol{G}}$ for any $\boldsymbol{a},\boldsymbol{b},\boldsymbol{e} \in \mathcal{M}$. Similarly, left or right invariance implies that distance is preserved under either left or right group operations.

with $\alpha, \beta \in \mathbb{R}^+$, are the only left-invariant metrics on SE(3)—i.e. metrics which hold under change of inertial frame. These are also the metrics of the manifold product $\mathrm{SO}(3) \times \mathbb{R}^3$.

The map that relates rigid-body displacements to screw motion is the matrix exponential. It relates elements of the Lie-algebra $\mathfrak{se}(3)$ to SE(3). Unlike the Riemannian exponential, the matrix exponential is not based on a metric but follows from the group structure of SE(3) [84]. Given a screw motion $\mathfrak{s}_{12}$, which describes the rigid-body displacement from $\boldsymbol{H}_1$ to $\boldsymbol{H}_2$, this raises the question: is there a left-invariant metric (2.28) which makes screw motion result in geodesics?

Figure 2.7a illustrates that screw motions do not generally lie on geodesics. If the screw motion that describes the rigid-body displacement from $\boldsymbol{H}_1$ to $\boldsymbol{H}_2$ would lie on a geodesic, there would not be a shorter path between $\boldsymbol{H}_1$ and $\boldsymbol{H}_2$. Yet, by introducing $\boldsymbol{H}_3$, we see that the combination of screw motion from $\boldsymbol{H}_1$ to $\boldsymbol{H}_2$ via $\boldsymbol{H}_3$ is shorter under any metric (2.28). Therefore, we can conclude that the matrix exponential does not map minimum distances on SE(3). Consequently, the matrix exponential and logarithmic maps are not suitable for our statistical framework which requires a minimum distance map.

Rigid-body transformations

The use of rigid-body transformations to represent end-effector pose is unnecessarily restrictive. It does not allow us to transform the positional and orientation parts of the end-effector pose using different transformations without violating the group structure. Figure 2.7b illustrates the need for different transformations. It shows how a taping motion across the seam of a carton box can be transformed to a box of different shape and pose. Here, the shape transforms the spatial part of the motion using an affine transformation, while the orientation part is transformed using a rotational transformation. The application of an affine transformation to an element of SE(3) does not necessary yield an element of of SE(3), as any scaling inflicted by the transformation would violate the orthonormal property of the SE(3) rotation. The way in which these affine transformations are applied in our framework is detailed in sections 2.5.5 and 4.3.1.

Coordinate dependence of TP-GMM

A celebrated benefit of SE(3) is the ability to represent rigid-body motion independent from an inertial frame. Yet, the context-adaptive approach, presented in Chapter 4 gains its generalization capabilities from a coordinate-dependent representation. This is conceptually illustrated in Figure 2.7c. The context-adaptive approach models motion in different local coordinate systems Ψ_i. Generalization to new context is achieved by placing the local representations in new context. To achieve this, the end-effector poses are defined in a coordinate-dependent way. A coordinate independent representation of rigid-body motion is thus not required in our application.

2.5. Gaussian Operations on Riemannian Manifolds

In this section the generalization of the Euclidean Gaussian to Riemannian manifolds is introduced. We will demonstrate how, and to what extend, this Riemannian Gaussian can generalize the properties of its Euclidean counterpart.

2.5.1. THE RIEMANNIAN GAUSSIAN

RIEMANNIAN CENTER OF MASS

The absence of a Euclidean structure makes the concept of a *mean value* not directly transferable to Riemannian manifolds. However, using the notion of variance $\sigma_{\mathrm{x}}(\boldsymbol{y}) = \mathbb{E}\left[\mathrm{dist}(\mathrm{x}, \boldsymbol{y})\right]$, we can define an alternative first mode that can be used in a Gaussian-like distribution.

Recall the definition of mean for a distribution $\mathcal{P}_{\mathrm{x}}$ of random variable x that is defined on a Euclidean space

$$\boldsymbol{\mu} = \mathbb{E}\left[\mathcal{P}_{\mathrm{x}}\right] = \int \boldsymbol{z}\, \mathcal{P}_{\mathrm{x}}(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z}, \tag{2.29}$$

where $\boldsymbol{\mu}, \boldsymbol{z} \in \mathbb{R}^d$. Its extension to Riemannian manifold is prevented by the integral over $\boldsymbol{z}$, as the sum over $\boldsymbol{z} \in \mathcal{M}$ is not defined. Variance, on the other hand, is the expectation of a real-valued distance function, and the resulting integral can be computed over the Riemannian manifold, namely

$$\sigma_{\mathrm{x}}(\boldsymbol{y}) = \mathbb{E}\left[\mathrm{dist}(\mathrm{x}, \boldsymbol{y})^2\right] = \int_{\mathcal{M}} \mathrm{dist}(\mathrm{x}, \boldsymbol{z})^2\, \mathcal{P}_{\mathrm{x}}(\boldsymbol{z})\, \mathrm{d}\mathcal{M}(\boldsymbol{z}). \tag{2.30}$$

with $\mathrm{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathcal{M}$, and $\mathrm{d}\mathcal{M}(\boldsymbol{z})$ an infinitesimal volume element (see [47]). The distance on the manifold corresponds to the length of the geodesics. We can thus use $\mathrm{dist}(\mathrm{x}, \boldsymbol{z})^2 = \mathrm{Log}_{\mathrm{x}}(\boldsymbol{z})^{\top} \mathrm{Log}_{\mathrm{x}}(\boldsymbol{z})$.

Fréchet showed that variance of a random variable is minimized for the mean value $\boldsymbol{\mu}$, i.e. $\boldsymbol{\mu} = \boldsymbol{y}$. The set of points that minimizes

$$\mathbb{E}\left[\mathrm{x}\right] = \arg\min_{\boldsymbol{y} \in \mathcal{M}} \left(\mathbb{E}\left[\mathrm{dist}(\mathrm{x}, \boldsymbol{y})^2\right]\right), \tag{2.31}$$

thus consists of mean points.

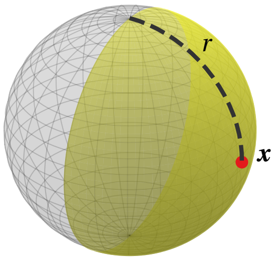


Figure 2.8: The ball $\mathcal{B}(\boldsymbol{x}, r) = \{\boldsymbol{y} \in \mathcal{M} \,|\, \mathrm{dist}(\boldsymbol{x}, \boldsymbol{y}) < r\}$, is said to be *geodesic* if it does not meet the cut locus of its center. This means that there exists a unique minimizing geodesic from the center to any point of the geodesic ball. The ball is said to be *regular* if its radius verifies $2r\sqrt{\kappa} < \pi$, where κ is the maximum of the Riemannian curvature in this ball. (definition adopted from [47]). The yellow cover visualizes the largest regular geodesic ball on $\mathcal{S}^2$. The ball is centered at center $\boldsymbol{x}$ and excludes the equator. It is geodesic on $\mathcal{S}^2$ because it does not cover the cut locus (the anti-podal of $\boldsymbol{x}$), and regular because its radius $r < \pi$ for a manifold curvature $\kappa = 1$.

The existence and uniqueness of the mean point is not guaranteed. Karcher and Kendall studied the uniqueness and existence of local minima of (2.30) [90, 91], which are called *Riemannian centers of mass* and sometimes (incorrectly) referred to as Karcher means [92]. They demonstrated that: (1) a distribution $\mathcal{P}_{\mathrm{x}}$ has a unique Riemannian center of mass when its support is included in a regular geodesic ball $\mathcal{B}(\boldsymbol{y}, r)$ (see Figure 2.8); and (2), (2.30) is convex if the support of $\mathcal{P}_{\mathrm{x}}$ is contained in $\mathcal{B}(\boldsymbol{y}, r)$, and the ball with double radius $\mathcal{B}(\boldsymbol{y}, r)$ is still geodesic and regular. These proofs have later been extended for manifolds with ψ-convexity to distributions with non-compact support, such as distributions of the exponential family.

Practically, (1) ensures that a sufficiently localized distribution $\mathcal{P}_{\mathrm{x}}$ is uni-modal, while (2) ensures that it can be found using gradient descent. Given the curvature of the Riemannian manifold, (1) allows us to determine if an estimated Riemannian center of mass is unique. This is the case if the observations $\{\boldsymbol{x}_i\}_n^N$ fit in a regular geodesic ball, i.e.

$$r = \max_i \big(\mathrm{dist}(\boldsymbol{\mu}, \boldsymbol{x}_i)\big) < \frac{\pi}{2\sqrt{\kappa}}, \tag{2.32}$$

with curvature κ.

Covariance

When the mean point(s), are determined. They can be used to compute the second moment: the covariance, the directional dispersion of the data. Within the Riemannian framework distance and direction of a point $\boldsymbol{x}$ with respect to another point $\boldsymbol{y}$ can be represented in the Euclidean tangent space $\mathcal{T}_{\boldsymbol{y}}\mathcal{M}$ using $\mathrm{Log}_{\boldsymbol{y}}(\boldsymbol{x})$, given that $\boldsymbol{x}$ lies within its domain $\mathcal{D}(\boldsymbol{y})$.

Since we have the ability to express distance and direction among points in a Euclidean fashion, we can compute the covariance similarly to Euclidean covariance, namely

$$\boldsymbol{\Sigma} = \mathbb{E}\left[\mathrm{Log}_{\boldsymbol{\mu}}(\mathrm{x})\,\mathrm{Log}_{\boldsymbol{\mu}}(\mathrm{x})^{\top}\right] = \int_{\mathcal{D}(\boldsymbol{\mu})} \mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{z})\,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{z})^{\top}\,\mathcal{P}_{\mathrm{x}}(\boldsymbol{z})\,\mathrm{d}\mathcal{M}(\boldsymbol{z}). \tag{2.33}$$

The Maximum Entropy Distribution

In [47], Pennec shows that the maximum entropy distribution given the first two moments—mean point and covariance—is a distribution of the exponential family

$$\mathcal{N}_{\mathcal{M}}\big(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda}\big) = k^{-1}\exp\left(-\frac{1}{2}\,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})^{\top}\,\boldsymbol{\Lambda}\,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})\right), \tag{2.34}$$

with k a normalization, $\boldsymbol{\mu} \in \mathcal{M}$ the Riemannian center of mass, and $\boldsymbol{\Lambda}$ the precision (also known as the concentration) defined in the tangent space $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$. This formulation of the Gaussian depends on the distance preserving mapping from the manifold to the tangent space ($\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})$) which allows the computation of Mahanalobis distance. This mapping provides us a way to describe coordination and variance among the dimensions of the manifold in the linear tensor $\boldsymbol{\Lambda}$. The potential of this tensor is demonstrated throughout this thesis: it enables regression on manifolds (Section 2.6), forms the basis of the synergetic controllers presented in Chapter 3, and describes weights of the Gaussian product that is used in chapters 4 and 5.

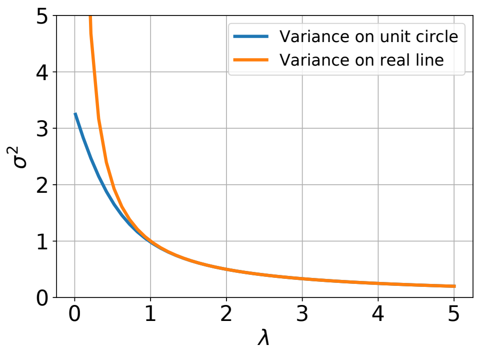


Figure 2.9: In the context of Riemannian manifolds, for sufficiently large precision, we can approximate variance to be inversely proportional to precision (i.e. $\mathbf{\Sigma} \approx \mathbf{\Lambda}^{-1}$). This is illustrated for the relation between variance and precision of a Gaussian on the unit circle $\mathcal{S}^1$ (blue), and the real line $\mathbb{R}$ (orange). For small precision, the true variance of the unit circle converges to a constant, while the real line follows the relation $\boldsymbol{\sigma}^2 = \boldsymbol{\lambda}^{-1}$. However, as precision increases, $\boldsymbol{\sigma}^2 = \boldsymbol{\lambda}^{-1}$ becomes a reasonable approximation for the distribution on the unit circle. This behavior generalizes to all compact manifolds [47].

The normalization coefficient k is given by

$$k = \int_{\mathcal{M}} \exp\left(-\frac{1}{2}\operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})^\top \mathbf{\Lambda} \operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})\right) \mathrm{d}\mathcal{M}(\boldsymbol{x}), \tag{2.35}$$

with $\mathrm{d}\mathcal{M}(\boldsymbol{x})$ an infinitesimal volume element (see Pennec *et al.* [47]), and the relation between covariance and precision

$$\Sigma = \int_{\mathcal{M}} \operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}) \operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})^\top \exp\left(-\frac{1}{2}\operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})^\top \mathbf{\Lambda} \operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})\right) \mathrm{d}\mathcal{M}(\boldsymbol{x}),. \tag{2.36}$$

The integrals in (2.35) and (2.36) require $\operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})$ to be continuous on $\mathcal{M}$. This is generally not the case; e.g. the spherical manifolds used in this work contain a discontinuity at the cut-locus. One can ignore this requirement at the cost of generating an underestimative model, whose error magnitude relates to the variance of the distribution. Simo-Serra *et al.* [51] demonstrated that an estimation error of over 1% on $\mathcal{S}^2$ requires a distribution with a standard deviation larger than 1.0 radian. Furthermore, (2.36) makes computation of concentration matrix from the precision matrix difficult. However, for reasonably small covariance, the precision can be approximated by $\mathbf{\Lambda} \approx \mathbf{\Sigma}^{-1}$. We illustrate this in Figure 2.9 for the special case of compact manifolds.

In this thesis, we rely on an approximation of the maximum entropy distribution for Riemannian Manifolds,

$$\mathcal{N}_{\mathcal{M}}(\boldsymbol{x}|\boldsymbol{\mu},\mathbf{\Lambda}) = \underbrace{\frac{1}{\sqrt{(2\pi)^d|\mathbf{\Sigma}|}}}_{k^{-1}} \exp\left(-\frac{1}{2}\operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})^\top \mathbf{\Sigma}^{-1} \operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})\right). \tag{2.37}$$

This approximation has also been used Simo-Serra *et al.* [51], Kim *et al.* [42] and Dubbelman [52]. It assumes the relation between the covariance and the precision to be $\mathbf{\Sigma} =$

	$\epsilon(\boldsymbol{\mu})$	$\boldsymbol{W}$	$\boldsymbol{J}$	$\boldsymbol{\Delta}$
MLE	$\begin{bmatrix}\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_n)\\ \vdots \\ \mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_n)\end{bmatrix}$	$\mathrm{diag}\begin{pmatrix}\boldsymbol{\Sigma}^{-1}\\ \vdots \\ \boldsymbol{\Sigma}^{-1}\end{pmatrix}$	$\begin{bmatrix}-\boldsymbol{I}\\ \vdots \\ -\boldsymbol{I}\end{bmatrix}$	$\frac{1}{\sum n^N h_n}\sum_{n=1}^{N} h_n \,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_n)$
Product	$\begin{bmatrix}-\mathrm{Log}_{\boldsymbol{\mu}}\big(\boldsymbol{\mu}_1\big)\\ \vdots \\ -\mathrm{Log}_{\boldsymbol{\mu}}\big(\boldsymbol{\mu}_P\big)\end{bmatrix}$	$\mathrm{diag}\begin{pmatrix}\boldsymbol{\Sigma}_{\Vert 1}^{-1}\\ \vdots \\ \boldsymbol{\Sigma}_{\Vert P}^{-1}\end{pmatrix}$	$\begin{bmatrix}-\boldsymbol{I}\\ \vdots \\ -\boldsymbol{I}\end{bmatrix}$	$\left(\sum_{p=1}^{P}\boldsymbol{\Sigma}_{\Vert p}^{-1}\right)^{-1}\sum_{p=1}^{P}\boldsymbol{\Sigma}_{\Vert p}^{-1}\,\mathrm{Log}_{\boldsymbol{\mu}}\big(\boldsymbol{\mu}_p\big)$
Condition	$\begin{bmatrix}\mathrm{Log}_{\boldsymbol{x}_{\mathcal{I}}}\big(\boldsymbol{\mu}_{\mathcal{I}}\big)\\ \mathrm{Log}_{\boldsymbol{x}_{\mathcal{O}}}\big(\boldsymbol{\mu}_{\mathcal{O}}\big)\end{bmatrix}$	$\boldsymbol{\Lambda}_{\Vert}$	$\begin{bmatrix}\boldsymbol{0}\\ -\boldsymbol{I}\end{bmatrix}$	$\mathrm{Log}_{\boldsymbol{x}_{\mathcal{O}}}\big(\boldsymbol{\mu}_{\mathcal{O}}\big)+\boldsymbol{\Lambda}_{\Vert\mathcal{O}\mathcal{O}}^{-1}\boldsymbol{\Lambda}_{\Vert\mathcal{O}\mathcal{I}}^{\top}\,\mathrm{Log}_{\boldsymbol{x}_{\mathcal{I}}}\big(\boldsymbol{\mu}_{\mathcal{I}}\big)$

Table 2.3: Overview of the parameters used in iterative likelihood maximization of the mean value presented in Section 2.5.

	MLE	Product	Condition
$\boldsymbol{\Sigma}$	$\frac{1}{\sum_i^N h_i}\sum_{n=1}^{N} h_i\,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_n)\,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_n)^{\top}$	$\left(\sum_{p=1}^{P}\boldsymbol{\Sigma}_{\Vert p}^{-1}\right)^{-1}$	$\boldsymbol{\Lambda}_{\Vert\mathcal{O}\mathcal{O}}^{-1}$

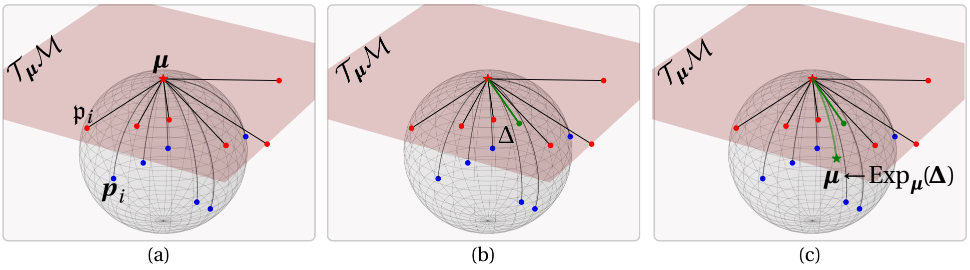
Table 2.4: Overview of the equations required to compute the covariance for the procedures presented in Section 2.5.

$\boldsymbol{\Lambda}^{-1}$, and the absence of a cut-locus, i.e. the domain of $\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}) = \mathbb{R}^d$. Throughout this thesis, we will refer to (2.37) as the (approximated) *Riemannian Gaussian*.

The approximations gain us simplicity, and efficiency at the cost of losing accuracy in the likelihood values for distributions with large variance. For position data these simplifications do not cause any error. For Euclidean data, (2.34) becomes the standard Euclidean Gaussian, and exactly matches (2.37).

For orientation data, the manifold used to express orientation data will influence the accuracy of our approximation. Throughout this thesis we rely on the unit quaternion to represent orientation. Its double covering of $SO(3)$ makes distributions of $SO(3)$ appear more concentrated on $\mathcal{S}^3$; a length on $SO(3)$ will appear halved on $\mathcal{S}^3$, and variance is thus be decreased by a factor 2^2 on $\mathcal{S}^3$. Furthermore, unit quaternions capture $SO(3)$ within a regular geodesic ball on $\mathcal{S}^3$, ensuring the existence of a unique mean (see also Section 2.5.1).

We find the presented approximation suited for our framework, because it allows us to generalize the operations used in state-of-the-art PbD methods in the Euclidean space. These applications typically involve movement generalization and synthesis, and do not heavily rely on (absolute) likelihood values.



$\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$
$\boldsymbol{\mu}$
$\mathfrak{p}_i$
$\boldsymbol{p}_i$
Δ
$\boldsymbol{\mu} \leftarrow \mathrm{Exp}_{\boldsymbol{\mu}}(\boldsymbol{\Delta})$
(a)
(b)
(c)

Figure 2.10: Visualization of iterative likelihood maximization required to find a Riemannian center of mass. (a) Given an initial guess of $\boldsymbol{\mu}$ (red star), the points $\boldsymbol{p}_i$ (blue dots) are projected in the tangent space $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$ using the logarithmic map $\mathrm{Log}_{\boldsymbol{\mu}}(\cdot)$ (red dots). (b) After projection, an update is computed, i.e. $\boldsymbol{\Delta} = \frac{1}{N}\mathfrak{p}_i$. (c) The update is projected back onto the manifold using the exponential map $\mathrm{Exp}_{\boldsymbol{\mu}}(\cdot)$ (green dot). After, the update steps (a-c) are repeated until $|\Delta|$ reaches a pre-defined convergence threshold.

2.5.2. ITERATIVE LIKELIHOOD MAXIMIZATION

The parameters of Riemannian Gaussian can be estimated by maximizing the likelihood of (2.37). The required procedure is similar for estimation based on empirical data, product of Gaussians, and Gaussian conditioning. In this section, we derive the general procedure for data-driven parameter estimation. This derivation follows Dubbelman [52]. Similar results can be found in [47, 90]. The special cases of product of Gaussians and Gaussian conditioning will be discussed in sections 2.5.3 and 2.5.4, respectively.

Given random points $\boldsymbol{x}_i \in \mathcal{M}$, our objective is to find $\boldsymbol{\mu}$, and $\boldsymbol{\Sigma}$ that maximize maximize (2.37). As common for distributions of the exponential family, this is most conveniently achieved by maximizing the log-likelihood,

$$L = c - \frac{1}{2}\sum_{i=1}^{N} h_i \,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_i)^{\top}\,\boldsymbol{\Sigma}^{-1}\,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_i)\,, \tag{2.38}$$

with c a constant, and h_i weights assigned to each point. Here, it is assumed that $h_i = 1$, but different weights are required when estimating the likelihood of a mixture of Gaussians (see Section 2.5.6).

Because $\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})$ is potentially non-linear, optimization of (2.34) requires an iterative procedure. This procedure first estimates the mean point, and then computes the covariance in the tangent space $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$. We illustrate the procedure in Figure 2.10, and describe its derivation in the remainder of this section.

First, the log-likelihood (2.38) is rearranged as

$$f(\boldsymbol{\mu}) = -\frac{1}{2}\epsilon(\boldsymbol{\mu})^{\top}\boldsymbol{W}\epsilon(\boldsymbol{\mu}), \tag{2.39}$$

where c is omitted as it is independent of the samples $\boldsymbol{x}_i$, and

$$\epsilon(\boldsymbol{\mu}) = \left[\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_0)^{\top}, \mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_1)^{\top}, \ldots, \mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_N)^{\top}\right]^{\top}, \tag{2.40}$$

is a stack of tangent space vectors in $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$. These vectors give the distance and direction of $\boldsymbol{x}_i$ with respect to $\boldsymbol{\mu}$. $\boldsymbol{W}$ is a weight matrix with a block diagonal structure (see Table 2.3).

We maximize (2.39) using a Gauss-Newton optimization. For this, we first make a second order Taylor expansion of (2.39), namely

$$f(\boldsymbol{\mu}) \approx f(\tilde{\boldsymbol{\mu}}) + \boldsymbol{\Delta}^{\top} f'(\tilde{\boldsymbol{\mu}}) + \frac{1}{2}\boldsymbol{\Delta}^{\top} f''(\tilde{\boldsymbol{\mu}})\Delta, \tag{2.41}$$

where

$$f' = \boldsymbol{J}^{\top}\boldsymbol{W}\epsilon(\tilde{\boldsymbol{\mu}}), \tag{2.42}$$
$$f'' = \boldsymbol{J}^{\top}\boldsymbol{W}\boldsymbol{J}, \tag{2.43}$$

are the gradient and the Hessian, respectively. $\boldsymbol{J}$ the Jacobian of $\epsilon(\boldsymbol{x})$ with respect to the tangent basis of $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$. It is a vertical concatenation of individual Jacobians $\boldsymbol{J}_i$ corresponding to $\text{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_i)$ which have the simple form $\boldsymbol{J}_i = -\boldsymbol{I}_d$.

From the Taylor expansion (2.41) we derive the Gauss-Newton update, which is

$$\boldsymbol{\Delta} = -\left(\boldsymbol{J}^{\top}\boldsymbol{W}\boldsymbol{J}\right)^{-1}\boldsymbol{J}^{\top}\boldsymbol{W}\epsilon(\boldsymbol{x}), \tag{2.44}$$

$\boldsymbol{\Delta}$ provides an estimate of the optimal value mapped into the tangent space $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$. The optimal value is obtained by mapping $\boldsymbol{\Delta}$ onto the manifold

$$\boldsymbol{\mu} \leftarrow \text{Exp}_{\boldsymbol{\mu}}(\boldsymbol{\Delta}). \tag{2.45}$$

The computation of (2.44) and (2.45) is repeated until $\boldsymbol{\Delta}$ reaches a predefined convergence threshold. We observed fast convergence in our experiments for Maximum Likelihood Estimate (MLE), Conditioning, Product, and GMR (typically 2-5 iterations). In the case of parameter estimation from data, the structure of $\boldsymbol{W}$ greatly simplifies $\boldsymbol{\Delta}$. This structure makes $\boldsymbol{\Delta}$ independent of the covariance $\boldsymbol{\Sigma}$. The covariance can thus be computed independently after $\boldsymbol{\mu}$ converged (see Table 2.4).

Note that the presented Gauss-Newton algorithm performs optimization over a domain that is a Riemannian manifold, while standard Gauss-Newton methods consider a Euclidean domain.

2.5.3. Gaussian Product

The log-likelihood of a product of Gaussians is given by

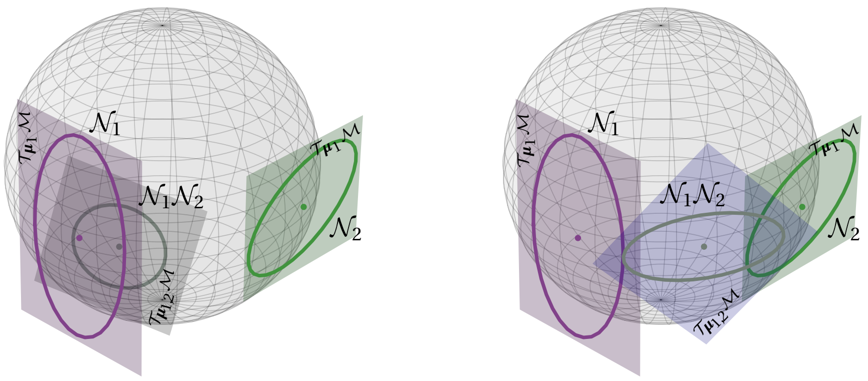
$$L(\boldsymbol{x}) = c - \frac{1}{2}\sum_{p=1}^{P}\text{Log}_{\boldsymbol{\mu}_p}(\boldsymbol{x})^{T}\boldsymbol{\Sigma}_p^{-1}\text{Log}_{\boldsymbol{\mu}_p}(\boldsymbol{x}), \tag{2.46}$$

where P represents the number of Gaussians to multiply, and $\boldsymbol{\mu}_p$ and $\boldsymbol{\Sigma}_p$ their parameters. Parameter estimation for the approximated Gaussian $\mathcal{N}\left(\tilde{\boldsymbol{\mu}},\tilde{\boldsymbol{\Sigma}}\right)$ can be achieved through likelihood maximization.

Ignoring the constant c, we can rewrite (2.46) into the form (2.39) by defining

$$\epsilon(\tilde{\boldsymbol{\mu}}) = \left[\text{Log}_{\boldsymbol{\mu}_1}\!\left(\tilde{\boldsymbol{\mu}}\right)^{\top}, \text{Log}_{\boldsymbol{\mu}_2}\!\left(\tilde{\boldsymbol{\mu}}\right)^{\top}, \ldots, \text{Log}_{\boldsymbol{\mu}_P}\!\left(\tilde{\boldsymbol{\mu}}\right)^{\top}\right]^{\top}, \tag{2.47}$$

and $\boldsymbol{W} = \text{diag}\left(\boldsymbol{\Sigma}_1^{-1}, \boldsymbol{\Sigma}_2^{-1}, \ldots, \boldsymbol{\Sigma}_P^{-1}\right)$, where $\tilde{\boldsymbol{\mu}}$ is the mean of the Gaussian we are approximating. Note that the vectors $\text{Log}_{\boldsymbol{\mu}_p}\!\left(\tilde{\boldsymbol{\mu}}\right)$ in (2.47) are not defined in the same tangent


$\mathcal{T}_{\mu_1}\mathcal{M}$
$\mathcal{N}_1$
$\mathcal{N}_1\mathcal{N}_2$
$\mathcal{T}_{\mu_1}\mathcal{M}$
$\mathcal{N}_2$
$\mathcal{T}_{\mu_{12}}\mathcal{M}$
$\mathcal{T}_{\mu_1}\mathcal{M}$
$\mathcal{N}_1$
$\mathcal{N}_1\mathcal{N}_2$
$\mathcal{T}_{\mu_1}\mathcal{M}$
$\mathcal{N}_2$
$\mathcal{T}_{\mu_{12}}\mathcal{M}$

(a) With parallel transport (b) Without parallel transport

Figure 2.11: In order to maximize the log-likelihood (2.46), the distance term (2.47) needs to be expressed in a common tangent space. Here, we visualize the outcome of the optimization with (a) and without (b) parallel transportation of the covariance. As the longitudal covariance of $\mathcal{N}_1$ is smaller than that of $\mathcal{N}_2$, the center of the product $\mathcal{N}_1\mathcal{N}_2$ should move towards $\mathcal{N}_1$. This is correctly achieved by the parallel transport of the covariance matrices.

space. Instead, they are defined in P different tangent spaces. In order to perform the likelihood maximization we need to switch the base and argument of Log() while ensuring that the original likelihood function (2.46) remains unchanged. This implies that the Mahalanobis distance should remain unchanged, i.e.

$$\mathrm{Log}_{\boldsymbol{\mu}_p}(\tilde{\boldsymbol{\mu}})^\top \boldsymbol{\Sigma}_p^{-1}\,\mathrm{Log}_{\boldsymbol{\mu}_p}(\tilde{\boldsymbol{\mu}}) = \mathrm{Log}_{\tilde{\boldsymbol{\mu}}}\big(\boldsymbol{\mu}_p\big)^\top \boldsymbol{\Sigma}_{\parallel p}^{-1}\,\mathrm{Log}_{\tilde{\boldsymbol{\mu}}}\big(\boldsymbol{\mu}_p\big), \tag{2.48}$$

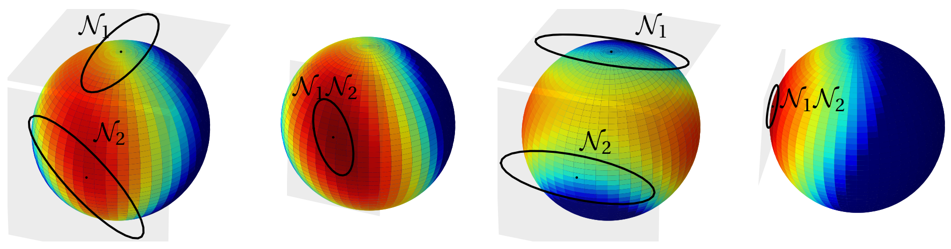
where $\boldsymbol{\Sigma}_{\parallel p}$ is a modified weight matrix that ensures an equal distance measure. It is computed through parallel transportation of $\boldsymbol{\Sigma}_p$ from $\boldsymbol{\mu}_p$ to $\tilde{\boldsymbol{\mu}}$ with

$$\boldsymbol{\Sigma}_{\parallel p} = \mathcal{A}_{\parallel\boldsymbol{\mu}_p}^{\tilde{\boldsymbol{\mu}}}(\boldsymbol{L}_p)^\top \mathcal{A}_{\parallel\boldsymbol{\mu}_p}^{\tilde{\boldsymbol{\mu}}}(\boldsymbol{L}_p), \tag{2.49}$$

where $\boldsymbol{L}_p$ is obtained through a symmetric decomposition of the covariance matrix, i.e. $\boldsymbol{\Sigma}_p = \boldsymbol{L}_p^\top \boldsymbol{L}_p$. This operation transports the eigencomponents of the covariance matrix [93]. Figure 2.11 compares the computation of the Gaussian product with and without parallel transport on $\mathcal{S}^2$.

Because parallel transport depends on the changing $\tilde{\boldsymbol{\mu}}$, (2.49) is evaluated at each iteration of the gradient descent. For spherical manifolds, parallel transport is the linear operation (2.12), and (2.49) simplifies to $\boldsymbol{\Sigma}_{\parallel p} = \boldsymbol{R}^\top \boldsymbol{\Sigma}_p \boldsymbol{R}$ with $\boldsymbol{R} = \mathcal{A}_{\parallel\boldsymbol{\mu}_p}^{\tilde{\boldsymbol{\mu}}}(\boldsymbol{I}_d)$. Using the transported covariances, both the gradient and the covariance can be computed. Their formula are presented in tables 2.3 and 2.4.

Equation (2.46) can have multiple extrema when $\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})$ is non-linear. As a result, the product of Gaussians on Riemannian manifolds is not guaranteed to be Gaussian. Figure 2.12 compares the true log-likelihood of a product of two Gaussians with the log-likelihood of a Gaussian approximation. The neighborhood in which the approximation



(a) (b)

Figure 2.12: Log-likelihood for the product of two Gaussians on the manifold $\mathcal{S}^2$. The Gaussians are visualized on their tangent spaces by the black ellipsoids. The color of the sphere corresponds to the value of the log-likelihood (high=red, low=blue). The true log-likelihood (equation (2.46) with $P = 2$) is displayed on the left of each subfigure, while the log-likelihood approximated by the product is displayed on the right. The configuration of the Gaussians in (a) results in a log-likelihood with a single mode, while (b) shows the special case of multiple modes. Note the existence of a saddle point to which the gradient descent could converge.

of the product by a single Gaussian is reasonable will vary depending on the values of $\boldsymbol{\mu}_p$ and $\boldsymbol{\Sigma}_p$. In chapters 4 we demonstrate that the approximation is suitable for movement regeneration in our context-adaptive framework.

The Gaussian product arises in different fields of probabilistic robotics. Generalizations of the Extended Kalman Filter [54] and the Unscented Kalman Filter [55] to Riemannian manifolds required a similar procedure. Similarly, Wolfe *et al.* proposed a similar procedure for Bayesian fusion on Lie-Groups [53].

2.5.4. GAUSSIAN CONDITIONING

In Gaussian conditioning, we compute the probability $\mathcal{P}\left(\boldsymbol{x}^{\mathcal{O}}|\boldsymbol{x}^{\mathcal{I}}\right) \sim \mathcal{N}(\boldsymbol{\mu}^{\mathcal{O}|\mathcal{I}}, \boldsymbol{\Sigma}^{\mathcal{O}|\mathcal{I}})$ of a Gaussian that encodes the joint probability density of $\boldsymbol{x}^{\mathcal{O}}$ and $\boldsymbol{x}^{\mathcal{I}}$. The log-likelihood of the conditioned Gaussian is given by

$$L(\boldsymbol{x}) = c - \frac{1}{2}\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})^{\top}\boldsymbol{\Lambda}\,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}), \tag{2.50}$$

with

$$\boldsymbol{x} = \begin{bmatrix}\boldsymbol{x}^{\mathcal{I}}\\ \boldsymbol{x}^{\mathcal{O}}\end{bmatrix}, \quad \boldsymbol{\mu} = \begin{bmatrix}\boldsymbol{\mu}^{\mathcal{I}}\\ \boldsymbol{\mu}^{\mathcal{O}}\end{bmatrix}, \quad \boldsymbol{\Lambda} = \begin{bmatrix}\boldsymbol{\Lambda}^{\mathcal{I}\mathcal{I}} & \boldsymbol{\Lambda}^{\mathcal{I}\mathcal{O}}\\ \boldsymbol{\Lambda}^{\mathcal{O}\mathcal{I}} & \boldsymbol{\Lambda}^{\mathcal{O}\mathcal{O}}\end{bmatrix}, \tag{2.51}$$

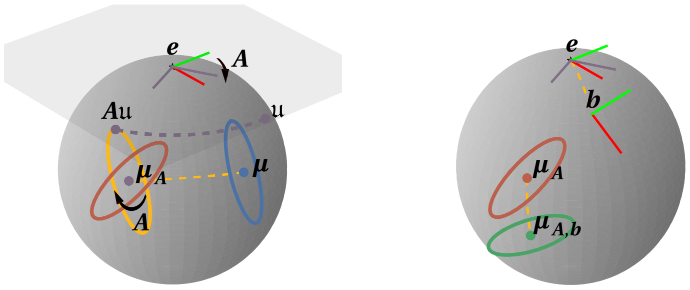
where the superscripts $\mathcal{O}$ and $\mathcal{I}$ indicate respectively the input and the output, and the precision matrix $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$ is introduced to simplify the derivation.

Equation (2.50) is in the form of (2.39). In the case of conditioning, we want to estimate $\boldsymbol{x}^{\mathcal{O}}$ given $\boldsymbol{x}^{\mathcal{I}}$ (i.e. $\boldsymbol{\mu}^{\mathcal{O}|\mathcal{I}}$). Similarly to the Gaussian product, we cannot directly optimize (2.50) because the dependent variable, $\boldsymbol{x}^{\mathcal{O}}$, is in the argument of the logarithmic map. This is again resolved by parallel transport, namely

$$\boldsymbol{\Lambda}_{\parallel} = \mathcal{A}_{\parallel}{}_{\boldsymbol{\mu}}^{\boldsymbol{x}}(\boldsymbol{V})^{\top}\mathcal{A}_{\parallel}{}_{\boldsymbol{\mu}}^{\boldsymbol{x}}(\boldsymbol{V}), \tag{2.52}$$

where $\boldsymbol{V}$ is obtained through a symmetric decomposition of the precision matrix: $\boldsymbol{\Lambda} = \boldsymbol{V}^{\top}\boldsymbol{V}$. Using the transformed precision matrix, the values for $\epsilon(\boldsymbol{x}_t)$ and $\boldsymbol{J}$ (both found in Table 2.3), we can apply (2.45) to obtain the update rule. The covariance obtained through Gaussian conditioning is given by $\boldsymbol{\Lambda}_{\parallel}^{-1}$. Note that we maximize the likelihood only with respect to $\boldsymbol{x}^{\mathcal{O}}$, and therefore $\mathbf{0}$ appears in the Jacobian listed in Table 2.3.

2.5.5. GAUSSIAN TRANSFORMATION



(a) Transformation $\boldsymbol{A}$ (b) Translation $\boldsymbol{b}$

Figure 2.13: Application of task-parameters $\boldsymbol{A} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{b} \in \mathcal{S}^2$ to a Gaussian defined on $\mathcal{S}^2$.

One of the challenges in PbD is to generalize skills to previously unseen situations, while keeping a small set of demonstrations. In task-parameterized representations [21], this challenge is tackled by considering the robot end-effector motion in different coordinate systems. These are defined in a global frame of reference through the *task parameters* $\boldsymbol{A}$ and $\boldsymbol{b}$, representing a linear transformation and translation, respectively. In Euclidean spaces this allows data to be projected to the global frame of reference through the linear operation $\boldsymbol{A}\boldsymbol{u}+\boldsymbol{b}$.

This linear operation can also be applied to the Riemannian Gaussian, namely

$$\boldsymbol{\mu}_{A,b} = \mathrm{Exp}_{\boldsymbol{b}}\big(\boldsymbol{A}\,\mathrm{Log}_{\boldsymbol{e}}(\boldsymbol{\mu})\big), \tag{2.53}$$

$$\boldsymbol{\Sigma}_{A,b} = (\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^{\top})_{\parallel_{\boldsymbol{b}}^{\boldsymbol{\mu}_{A,b}}}, \tag{2.54}$$

with $(\cdot)_{\parallel_{\boldsymbol{b}}^{\boldsymbol{\mu}_{A,b}}}$ the parallel transportation of the covariance matrix from $\boldsymbol{b}$ to $\boldsymbol{\mu}_{A,b}$. These operations are used in Chapter 4 to generalize the task-parameterized framework to Riemannian manifolds.

We illustrate the individual effect of $\boldsymbol{A}$ and $\boldsymbol{b}$ in Figure 2.13. The rotation $\boldsymbol{A}$ is expressed in the tangent space of $\boldsymbol{e}$. Each center $\boldsymbol{\mu}$ of the Gaussian is expressed as $\boldsymbol{u} = \mathrm{Log}_{\boldsymbol{e}}(\boldsymbol{\mu})$ in this tangent space. After applying the rotation $\boldsymbol{u}' = \boldsymbol{A}\boldsymbol{u}$, and by exploiting the property of homogeneous space in (2.9), the result is moved to $\boldsymbol{b}$ with $\mathrm{Exp}_{\boldsymbol{b}}(\boldsymbol{u}') = \mathcal{A}_{\boldsymbol{e}}^{\boldsymbol{b}}\big(\mathrm{Exp}_{\boldsymbol{e}}(\boldsymbol{u}')\big)$, see (2.53). To maintain the relative orientation between the local frame and the covariance, we parallel transport the covariance $\boldsymbol{\Sigma}_{A}$ from the $\boldsymbol{b}$ to $\boldsymbol{\mu}_{A,b}$ and obtain $\boldsymbol{\Sigma}_{A,b}$. The transport compensates the tangent space misalignment caused by moving the local frame away from the origin. Figure 2.13b shows the application of $\boldsymbol{b}$ to the result of Figure 2.13a.

2.5.6. GAUSSIAN MIXTURE MODEL AND EXPECTATION MAXIMIZATION (EM)

Similarly to a GMM in Euclidean space, a GMM on a Riemannian manifold is defined by a weighted sum of Gaussians

$$\mathcal{P}(\boldsymbol{x}) = \sum_{i=1}^{K} \pi_i \mathcal{N}\left(\boldsymbol{x}; \boldsymbol{\mu}_\mathrm{i}, \boldsymbol{\Sigma}_\mathrm{i}\right),$$

where π_i are the priors, with $\sum_i^K \pi_i = 1$, $\pi_i \geq 0$. In PbD, they are used to represent non-linear movements in a probabilistic manner. Parameters of the GMM can be estimated by EM. This is an iterative process that soft-clusters the data (Expectation step), and subsequently updates the Gaussian parameters using a weighted MLE (Maximization step) [94]. The concept of EM is applicable to data defined in a Riemannian manifold, as previously demonstrated by Simo-Serra *et al.* [51].

Algorithm 2.1 describes EM for Riemannian data. The inputs of EM are the N data points $\boldsymbol{x}_{1:N} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, and an initial estimate of the GMM parameters. Various methods exist to initialize the parameters. In Appendix A.1 we describe two common approaches: K-means and K-bins.

The core of EM consists of two elements: the Expectation step (E-step, lines 3–7) and the Maximization (M-step, lines 9–13). The E-step computes the weights $\gamma_{n,k}$, which are called responsibilities as they describe how responsible each Gaussian is for a data point. During the M-step the current estimate of the GMM parameters is updated based on a weighted average of the data points. For each state k, the M-step first computes the mean (line 11). This involves the likelihood maximization (2.38) with the weights $h_i = \gamma_{n,k}$ $(i = n)$. Using the estimated mean $\boldsymbol{\mu}_k$, the covariance $\boldsymbol{\Sigma}_k$ is computed (line 12). Here, N_k normalizes the weights in such a way that $\sum_n^N \gamma_{n,k} = 1$. The prior π_k describes the Gaussian's overall responsibility for the data. Finally, the log-likelihood of the GMM is computed to assess the change δll between two iterations. If the log-likelihood increase is smaller than a pre-defined convergence threshold the parameters are assumed to have converged. A low convergence threshold results in a more accurate solution at the cost of more iterations. In this thesis, an empirically chosen value of $1e-5$ is used. The main difference between EM for Euclidean data and EM for Riemannian data, is the need of a Gauss-Newton optimization to compute the mean (line 11). Similarly, to MLE of a single Gaussian the uniqueness of $\boldsymbol{\mu}_k$ is not guaranteed. In practice, the Gaussians of a GMM are sufficiently concentrated and converge to a unique Riemannian center of mass (see also Section 2.5.1).

2.5.7. GAUSSIAN MIXTURE REGRESSION

A popular regression technique for Euclidean GMM is Gaussian Mixture Regression (GMR) [21]. It approximates the conditioned GMM using a single Gaussian, i.e.

$$\mathcal{P}\left(\boldsymbol{x}^{\mathcal{O}} | \boldsymbol{x}^{\mathcal{I}}\right) \approx \mathcal{N}\left(\hat{\boldsymbol{\mu}}^{\mathcal{O}}, \hat{\boldsymbol{\Sigma}}^{\mathcal{O}}\right).$$

Algorithm 2.1 Expectation Maximization for GMM

```
1: function EM_GMM($\boldsymbol{x}_{1:N}$, $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k\}_{k=1}^K$)
2:     while $\delta ll > \epsilon_{conv}$ do          ▷ Check change of likelihood, $\delta ll$, between iterations
3:         # E-Step:
4:         for $k \in \{1, \cdots, K\}$ do
5:             for $n \in \{1, \cdots, N\}$ do
6:                 $\gamma_{n,k} = \frac{\pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{i=1}^K \pi_i \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}$
7:             $N_k = \sum_n^N \gamma_{n,k}$
8:
9:         # M-Step:
10:        for $k \in \{1, \cdots, K\}$ do
11:            $\boldsymbol{\mu}_k \leftarrow \arg\max_{\boldsymbol{\mu}_k} \left(\mathrm{L}\left(\boldsymbol{x}_{1:N}, \boldsymbol{\mu}_k, \gamma_{1:N,k}\right)\right)$
12:            $\boldsymbol{\Sigma}_k \leftarrow \frac{1}{N_k} \sum_n^N \gamma_{n,k} \mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_n) \mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_n)^\top$
13:            $\pi_k \leftarrow \frac{N_k}{N}$
14:
15:        # Compute log-likelihood:
16:        $ll = \sum_{n=1}^N \ln\left(\sum_{k=1}^K \pi_k \mathcal{N}\left(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)\right)$
17:    return $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k\}_{k=1}^K$
```

In Euclidean space, the parameters of this Gaussian are formed by a weighted sum of the expectations and covariances, namely,

$$\hat{\boldsymbol{\mu}}^{\mathcal{O}} = \sum_{i=1}^{K} h_i \, \mathbb{E}\left[\mathcal{N}\left(\boldsymbol{x}^{\mathcal{O}}|\boldsymbol{x}^{\mathcal{I}}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)\right], \tag{2.55}$$

$$\hat{\boldsymbol{\Sigma}}^{\mathcal{O}} = \sum_{i=1}^{K} h_i \, \mathrm{cov}\left[\mathcal{N}\left(\boldsymbol{x}^{\mathcal{O}}|\boldsymbol{x}^{\mathcal{I}}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)\right], \tag{2.56}$$

$$\text{with} \quad h_i = \frac{\mathcal{N}\left(\boldsymbol{x}^{\mathcal{I}}; \boldsymbol{\mu}_i^{\mathcal{I}}, \boldsymbol{\Sigma}_i^{\mathcal{I}\mathcal{I}}\right)}{\sum_{j=1}^K \mathcal{N}\left(\boldsymbol{x}^{\mathcal{I}}; \boldsymbol{\mu}_j^{\mathcal{I}}, \boldsymbol{\Sigma}_j^{\mathcal{I}\mathcal{I}}\right)}. \tag{2.57}$$

We cannot directly apply (2.55) and (2.56) to a GMM defined on a Riemannian manifold. First, because the computation of (2.55) would require a weighted sum of manifold elements—an operation that is not available on the manifold. Secondly, because directly applying (2.56) would incorrectly assume the alignment of the tangent spaces in which $\boldsymbol{\Sigma}_i$ are defined.

To remedy the computation of the mean, we employ the approach for MLE given in Section 2.5.2, and compute the weighted mean iteratively in the tangent space $\mathcal{T}_{\hat{\boldsymbol{\mu}}^{\mathcal{O}}}\mathcal{M}$, namely

$$\boldsymbol{\Delta} = \sum_{i=1}^{K} h_i \, \mathrm{Log}_{\hat{\boldsymbol{\mu}}^{\mathcal{O}}}\left(\mathbb{E}[\mathcal{N}\left(\boldsymbol{x}^{\mathcal{O}}|\boldsymbol{x}^{\mathcal{I}}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)]\right), \tag{2.58}$$

$$\hat{\boldsymbol{\mu}}^{\mathcal{O}} \leftarrow \mathrm{Exp}_{\hat{\boldsymbol{\mu}}^{\mathcal{O}}}(\boldsymbol{\Delta}). \tag{2.59}$$

Here, $\mathbb{E}[\mathcal{N}(\boldsymbol{x}^{\mathcal{O}}|\boldsymbol{x}^{\mathcal{I}};\boldsymbol{\mu}_{\mathrm{i}},\boldsymbol{\Sigma}_{\mathrm{i}})]$ refers to the conditional mean point obtained using the procedure described in Section 2.5.4. Note that the computation of the responsibilities h_i is straightforward using the Riemannian Gaussian (2.34).

After convergence of the mean, the covariance is computed in the tangent space defined at $\hat{\boldsymbol{\mu}}^{\mathcal{O}}$. First, $\boldsymbol{\Sigma}_i$ are parallel transported from $\mathcal{T}_{\boldsymbol{\mu}_i}\mathcal{M}$ to $\mathcal{T}_{\hat{\boldsymbol{\mu}}^{\mathcal{O}}}\mathcal{M}$, and then summed similarly to (2.56) with

$$\hat{\boldsymbol{\Sigma}}^{\mathcal{O}} = \sum_{i}^{K} h_i\, \boldsymbol{\Sigma}_{\parallel i}, \text{ where} \tag{2.60}$$

$$\boldsymbol{\Sigma}_{\parallel i} = \mathcal{A}_{\parallel \boldsymbol{\mu}_i}^{\hat{\boldsymbol{\mu}}^{\mathcal{O}}}(\boldsymbol{L}_i)^{\top}\, \mathcal{A}_{\parallel \boldsymbol{\mu}_i}^{\hat{\boldsymbol{\mu}}^{\mathcal{O}}}(\boldsymbol{L}_i), \tag{2.61}$$

and $\boldsymbol{\Sigma}_i = \boldsymbol{L}_i\boldsymbol{L}_i^{\top}$.

2.6. APPLICATION

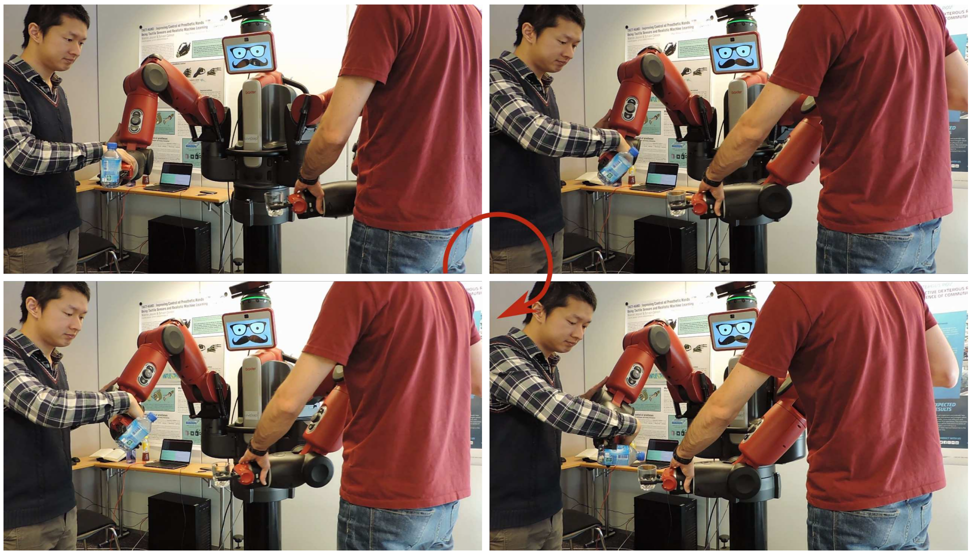
The Cartesian product allows us to combine a variety of Riemannian manifolds. Rich behavior can be encoded on such manifolds using relatively simple statistical models. We demonstrate this by encoding a bi-manual pouring skill with a single multivariate Riemannian Gaussian, and reproduce it using the presented Gaussian conditioning.

We transfer the pouring task using 3 kinesthetic demonstrations on 6 different locations in Baxter's workspace while recording the two end-effector poses (positions and orientations, 18 demonstrations in total). The motion consists of an adduction followed by an abduction of both arms, while holding the left hand horizontal and tilting the right hand (left and right seen from the robot perspective). Snapshots of a demonstration from two users (each moving one arm) are shown in Figure 2.14 and the typical demonstrations are shown in the video. The demonstrated data lie on the Riemannian manifold $\mathbb{R}^3 \times \mathcal{S}^3 \times \mathbb{R}^3 \times \mathcal{S}^3$. We encode the demonstrations in a single Gaussian defined on this manifold, i.e. we assume the recorded data to be distributed as $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$, with $\boldsymbol{x} = (\boldsymbol{x}_L, \boldsymbol{q}_L, \boldsymbol{x}_R, \boldsymbol{q}_R)$ composed of the position and quaternion of the left and right end-effectors.

Figure 2.15a shows the mean pose and its corresponding covariance, which is defined in the tangent space $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{B}$. The x_1, x_2 and x_3 axes are displayed in red, blue and green, respectively. The horizontal constraint of the left hand resulted in low rotational variance around the x_2 and x_3 axes, which is reflected in the small covariance at the tip of the x_1 axis. The low correlation of its x_2 and x_3 axes with other variables confirms that the constraint is properly learned (Figure 2.15b).

The bi-manual coordination required for the pouring task is encoded in the correlation coefficients of the covariance matrix[5] visualized in Figure 2.15b. The block-diagonal elements of the correlation matrix relate to the correlations within the manifolds, and the off-diagonal elements indicate the correlations between manifolds. Strong positive and negative correlations appear in the last block column/row of the correlation matrix. The strong correlation in the rotation of the right hand (lower right corner of the matrix) confirms that the main action of rotation is around the x_3-axis (blue) which causes the

[5]We prefer to visualize the correlation matrix, which only contains the correlation coefficients, because it highlights the coordination among variables.


(a) Typical demonstration

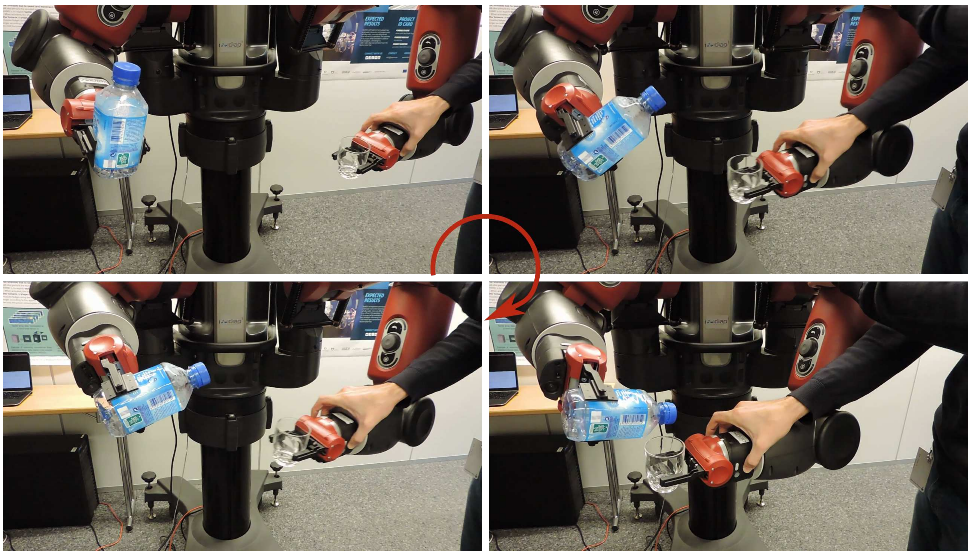
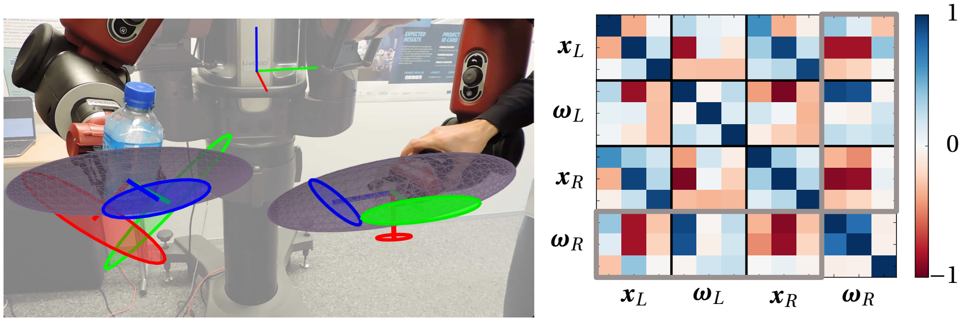

(b) Typical reproduction

Figure 2.14: Snapshot sequences of a typical demonstration (top) and reproduction (bottom) of the pouring task.



(a) End-effector covariance (b) End-effector correlation

Figure 2.15: The encoded task projected on the left and right end-effectors. (a) Gray ellipsoids visualize the spatial covariance (i.e. $\Sigma_{\boldsymbol{x}_L}$, $\Sigma_{\boldsymbol{x}_R}$), their centers are the mean end-effector positions ($\boldsymbol{\mu}_{x_L}$, $\boldsymbol{\mu}_{x_R}$). Each set of colored orthogonal lines depicts the end-effector mean orientation, its covariance is visualized by the ellipses at the end of each axis. (b) Correlation encoded within and between the different manifolds. The gray rectangles mark the correlations required for the reproduction presented in this experiment.

x_1 (red) and x_2 (green) axes to move in synergy. The strong correlations of the position $\boldsymbol{x}_L$, $\boldsymbol{x}_R$, and rotation $\boldsymbol{\omega}_L$ with rotation $\boldsymbol{\omega}_R$, demonstrate their importance in the task.

These correlations can be exploited to create a controller which can adapt online to new situations. To test the responsive behavior, we compute the causal relation $\mathcal{P}\left(\boldsymbol{q}_R|\boldsymbol{x}_R, \boldsymbol{q}_L, \boldsymbol{x}_L\right)$ through the Gaussian conditioning approach presented in Sec. 2.5.4. A typical reproduction is shown in Figure 2.14, and others can be found in the corresponding video. In contrast to the original demonstrations, which show noisy synchronization patterns from one recording to the next, the reproduction shows a smooth coordination behavior. The orientation of the right hand correctly adapts to the position of the right hand and the pose of the left hand. This behavior generalizes outside the demonstration area.

To assess the regression quality, we perform a cross-validation on 12 out of the 18 original demonstrations (2 demonstrations on 6 different locations).The demonstrations are split in training set of 8 and a validation set of 4 demonstrations, yielding $\binom{12}{8} = 495$ combinations.[6] For each of the N data points $\left(\boldsymbol{x}_L, \boldsymbol{q}_L, \boldsymbol{x}_R, \boldsymbol{q}_R\right)$ in the validation set, we compute the average rotation error between the demonstrated right hand rotation $\boldsymbol{q}_R$, and the estimated rotation $\hat{\boldsymbol{q}}_R = \mathbb{E}[\mathcal{P}\left(\boldsymbol{q}_R|\boldsymbol{x}_R, \boldsymbol{q}_L, \boldsymbol{x}_L\right)]$, i.e. the error statistic is $\frac{1}{N}\sum_i \|\operatorname{Log}_{\hat{\boldsymbol{q}}_{R,i}}\left(\boldsymbol{q}_{R,i}\right)\|$. The results of the cross-validation are summarized in the boxplot of Figure 2.16. The median rotation error is about 0.15 radian, and the interquartile indicates a small variance among the results. The far outliers (errors in the range 0.5 – 0.8 radian) correspond to combinations in which the workspace areas covered by the training set and validation set are disjoint. Such combinations make generalization harder, yielding the relatively large orientation error.

[6]The number of combinations to assess in cross-validation quickly grows with the number of demonstrations. To limit the computational time of cross-validation, only a subset of demonstrations is used in cross-validation.

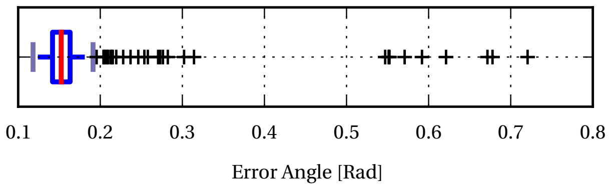


Figure 2.16: Cross validation results described in Section 2.6 .

2.7. DISCUSSION

In this chapter, we showed how GMM-based methods for PbD can be extended to Riemannian manifolds. We described how to perform Gaussian conditioning, Gaussian product and Gaussian transformations on Riemannian manifolds. These are the elementary tools required to extend GMM-GMR and TP-GMM as will be demonstrated in Chapter 4.

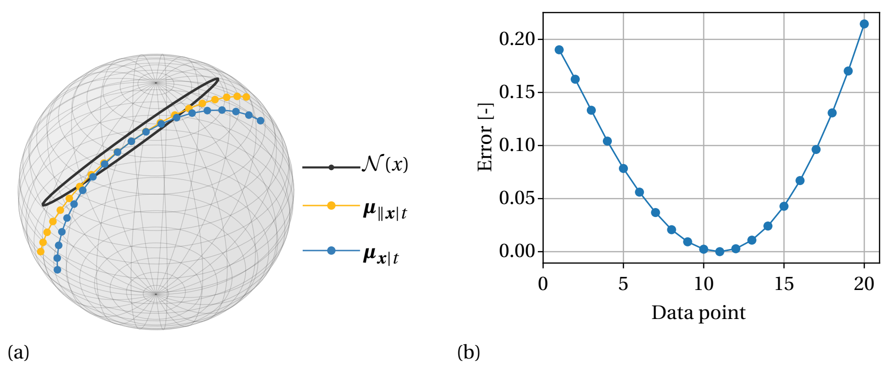


Figure 2.17: Visualization of Gaussian conditioning with and without parallel transport, by computing $\mathcal{N}(\boldsymbol{x}|t)$ with $\boldsymbol{x} \in \mathcal{S}^2$, $t \in \mathbb{R}$. (a) shows the output manifold where the marginal distribution $\mathcal{N}(\boldsymbol{x})$ is displayed in gray. (b) shows, per data point, the distance between conditioning with and without parallel transport. See Section 2.7 for further details.

Both Simo-Serra *et al.* [51] and Kim *et al.* [42] follow a Riemannian approach, although not explicitly mentioned by Kim *et al.* . Similar to our work, they rely on a simplified version of the maximum entropy distribution on Riemannian manifolds [47]. Independently, they present a method for Gaussian conditioning. But the proposed methods do not properly generalize the linear behavior of Gaussian conditioning to Riemannian manifolds, i.e. the means of the conditioned distribution do not lie on a single geodesic—the generalization of a straight line on Riemannian manifolds. This is illustrated in Figure 2.17. Here, we computed the update $\boldsymbol{\Delta}$ (given in Table 2.3 row 3, column 4) for Gaussian conditioning $\mathcal{N}(\boldsymbol{x}|t)$ with and without parallel transport, i.e. using $\boldsymbol{\Lambda}$ and $\boldsymbol{\Lambda}_{\|}$, respectively. Without parallel transport the regression output (solid blue line) does not lie on a geodesic, since it does not coincide with the (unique) geodesic between the outer points (dotted blue line). Using the proposed method that relies on the parallel transported precision matrix $\mathbf{\Lambda}_{\parallel}$, the conditioned means $\boldsymbol{\mu}_{\parallel \boldsymbol{x}|t}$ (displayed in yellow) follow a geodesic path on the manifold, thus generalizing the 'linear' behavior of Gaussian conditioning to Riemannian manifolds.

The manifold of unit quaternions is very regular as it has a constant curvature. Euclidean methods (e.g. [41]) can handle such data when aided by regularization heuristics. In fact, as such methods do not require an iterative solver, they are computationally more efficient than the Riemannian approach. What is the added benefit of the Riemannian approach compared to the Euclidean approach? First, the Riemannian approach omits the need for normalization heuristics, making it more generic. Furthermore, it provides a proper way of defining covariance: the Euclidean method provides a 4 dimensional covariance matrix even-though there are only 3 rotational degrees of freedom; the Riemannian approach properly takes the geometry into account, and estimates the covariance matrix in a 3 dimensional Euclidean tangent space. The results presented in Section 3.5 indicate that such covariance structure is, in some cases, better able to extract synergies from demonstration data. Finally, the Riemannian approach outperforms the Euclidean approach when handling more complex manifolds such as SPD matrices [49,50] by using a unified methodology that can be applied on various manifolds.

3

Learning Synergetic Control

3.1. Introduction

Synergies are functional groupings of elements that are constrained to work as a single unit [16]. Their existence potentially explains how biological sensorimotor systems are able to perform tasks accurately, despite high redundancies and noisy sensors [12, 13, 16, 96]. Synergies also appear at task level: grasping and manipulation require coordination between objects in the environment and the degrees of freedom of the hands. Similarly, synergies can be used as elementary units of behavior in the context of robotic control. By specifying task-dependent coordination at a low control level, one can achieve task specific disturbance rejection. In this chapter, we show that the covariance information of a Riemannian Gaussian can be used to derive synergetic controllers. It demonstrates that the Riemannian approach is able to encode and synthesize synergetic behavior.

The proposed control approach relies on the Linear Quadratic Regulator (LQR)—a control paradigm that simplifies the design of optimal controllers for linear dynamical systems. This optimal regulator is found by minimizing a cost function. This quadratic function is parameterized by a tracking cost matrix $\boldsymbol{Q}$ and a control cost matrix $\boldsymbol{R}$. Often, the tracking cost is manually defined using a diagonal $\boldsymbol{Q}$ matrix, thereby ignoring potential functional relations among state variables. Instead, we propose to relate the tracking cost to the covariance information of the Gaussian, whose structure includes such functional couplings. The ability to specify synergies through $\boldsymbol{Q}$ while guaranteeing stability makes the LQR an ideal method for our approach.

Task-space synergies require a suitable parameterization of robot pose, which involves both position and orientation. Since a global, singularity free, Euclidean representation of orientation does not exist [68], common methods available for Programming by Demonstration (PbD) and LQR are not directly applicable. We build upon the probabilistic framework for PbD on Riemannian manifolds introduced in Chapter 2. This framework allows us to learn distributions over robot poses whose support is contained

The contents of this chapter have been published in the proceedings of IEEE/ISR International Conference on Intelligent Robotics and Systems [95].

in a regular geodesic ball [47]. In practice, this restricts the orientation data to lie within a $\pm\pi$ radius of the empirical mean (the Riemannian center of mass). This is achieved by encoding robot poses as elements on the manifold $\mathbb{R}^3 \times \mathcal{S}^3$—The Cartesian product of the 3-dimensional Euclidean space and the unit-quaternion manifold $\mathcal{S}^3$, respectively.

The contributions of this chapter are two-fold: i) We demonstrate that the Riemannian Gaussian can be used to encode rich synergies of task-space manipulation that involve position and orientation; ii) We show how infinite horizon LQR can be used to regulate synergies that are defined on Riemannian manifolds.

The remainder of this chapter is structured as follows. Section 3.2 discusses previous work related to LQR on non-Euclidean spaces and controller-gain estimation based on demonstration data. Then, we introduce our method for LQR on Riemannian manifolds in Section 3.3. The efficacy of the approach is shown through an experimental evaluation involving bi-manual synergy transfer in Section 3.4. Finally, we will discuss some intrinsic geometric limitations of control on manifolds in Section 3.5.

3.2. RELATED WORK

Different approaches for LQR on $SO(3)$, $SE(3)$ or coverings of these groups exist. Saccon *et al.* [97] derive a LQR controller on $SO(3)$ through Pontryagin's Maximum Principle. Marinho *et al.* [98] use a dual-quaternion representation to derive a LQR tracking controller. The latter involves converting the dual quaternion transformation-invariant error into an affine time-varying system. Such representation can be compared to a pose manifold, yet the position quaternion does not represent well the Cartesian space, and the method requires the manual specification of the control and state error costs. Similarly, Wang and Yu [99] present a dual quaternion controller for rigid-body motion stabilization and tracking, built on a screw theory formulation.

Our approach to learn synergies from demonstration involves the estimation of stiffness and damping matrices from the correlation observed in the demonstration data. Similarly, Rozo *et al.* [100] and Saveriano and Lee [101] estimate the stiffness directly from the covariance information. Smoother stiffness profiles can be obtained from the covariance information through LQR as demonstrated by Medina *et al.* [36], Calinon *et al.* [18] and Zeestraten *et al.* [19]. Kronander and Billard [102] use a combination of tactile and kinesthetic teaching to communicate the desired stiffness of the robot along a trajectory. Unlike these previous works, the presented method considers coordination among position and orientation of multiple end-effectors.

3.3. LQR IN THE TANGENT SPACE

We start with a training set consisting of N data points, $\boldsymbol{x} \in \mathcal{M}$. This set potentially contains synergetic coupling among the manifold dimensions. Our aim is to find a controller that preserves these synergies. To identify them, we estimate the center $\boldsymbol{\mu} \in \mathcal{M}$ and covariance $\boldsymbol{\Sigma} \in \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$ of a Riemannian Gaussian using the Maximum Likelihood Estimate (MLE) (see Section 2.5.2). The covariance matrix encodes the local synergies around the estimated center. Similarly to previous work [18, 19, 36], we use LQR to replicate the encoded behavior. LQR is a controller for linear systems of the form $\dot{\boldsymbol{\xi}} = \boldsymbol{A}\boldsymbol{\xi} + \boldsymbol{B}\boldsymbol{u}$

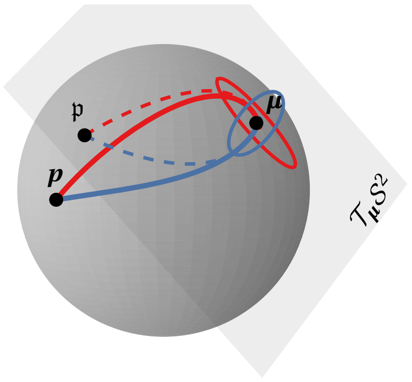


Figure 3.1: Visualization of state evolution obtained by Riemannian LQR on the system state manifold, $\mathcal{M}_{\boldsymbol{s}} = \mathcal{S}^2 \times \mathbb{R}^2$, for two different covariance matrices (red and blue ellipses). The initial state of the system is indicated by $\boldsymbol{p}$ and the desired state by $\boldsymbol{\mu}$. a) The figure shows the response path in $\mathcal{S}^2$ and $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{S}^2$. The response on the manifold is visualized by the solid lines, and the response on the tangent space by the dotted lines of corresponding color.

that optimizes a cost function that is quadratic in both state $\boldsymbol{\xi}$ and control input $\boldsymbol{u}$,

$$c = \frac{1}{2}\int \left(\boldsymbol{\xi}^{\top}\boldsymbol{Q}\boldsymbol{\xi} + \boldsymbol{u}^{\top}\boldsymbol{R}\boldsymbol{u}\right)\mathrm{d}t. \tag{3.1}$$

The solution to this optimal control problem is a state-feedback controller of the form $\boldsymbol{u} = \boldsymbol{L}\boldsymbol{\xi}$. Its gain matrix, $\boldsymbol{L}$, is obtained by solving an algebraic Riccati equation (see e.g. [103]).

The required linear system cannot be defined on the manifold, since it is not a vector space. However, we can exploit the linear tangent spaces to achieve a similar result. The state error between the desired state $\boldsymbol{p}_d$ and current state $\boldsymbol{p}$ can be computed using the logarithmic map $\boldsymbol{\mathfrak{e}} = \mathrm{Log}_{\boldsymbol{p}_d}(\boldsymbol{p})$ that projects the minimum length path between $\boldsymbol{p}_d$ and $\boldsymbol{p}$ into the Euclidean space $\mathcal{T}_{\boldsymbol{p}_d}\mathcal{M}$. We define the linear time-invariant system,

$$\underbrace{\begin{bmatrix}\dot{\boldsymbol{\mathfrak{e}}}\\ \ddot{\boldsymbol{\mathfrak{e}}}\end{bmatrix}}_{\dot{\boldsymbol{\xi}}} = \underbrace{\begin{bmatrix}\boldsymbol{0} & \boldsymbol{I}\\ \boldsymbol{0} & -\boldsymbol{M}^{-1}\boldsymbol{C}\end{bmatrix}}_{\boldsymbol{A}}\underbrace{\begin{bmatrix}\boldsymbol{\mathfrak{e}}\\ \dot{\boldsymbol{\mathfrak{e}}}\end{bmatrix}}_{\boldsymbol{\xi}} + \underbrace{\begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{M}^{-1}\end{bmatrix}}_{\boldsymbol{B}}\boldsymbol{\mathfrak{u}}. \tag{3.2}$$

with inertia matrix $\boldsymbol{M}$ and damping matrix $\boldsymbol{C}$. The augmentation with $\dot{\boldsymbol{\mathfrak{e}}}$ makes the system state manifold to be $\mathcal{M}_{\boldsymbol{s}} = \mathcal{M} \times \mathbb{R}^d$. As a result, the linear system is defined in $\mathcal{T}_{\bar{\boldsymbol{p}}_d}\mathcal{M}_{\boldsymbol{s}}$ with $\bar{\boldsymbol{p}}_d \in \mathcal{M}_{\boldsymbol{s}}$ (the original desired state augmented with a desired velocity), and the mapping from the manifold to this tangent space

$$\boldsymbol{\xi} = \mathrm{Log}_{\bar{\boldsymbol{p}}_d}\left(\bar{\boldsymbol{p}}\right). \tag{3.3}$$

The covariance $\boldsymbol{\Sigma}$ of a Riemannian Gaussian $\mathcal{N}_{\mathcal{M}}(\boldsymbol{\mu},\boldsymbol{\Sigma})$ describes the variance and correlation of the state variables in a tangent space defined at $\boldsymbol{\mu} \in \mathcal{M}$. By assuming that

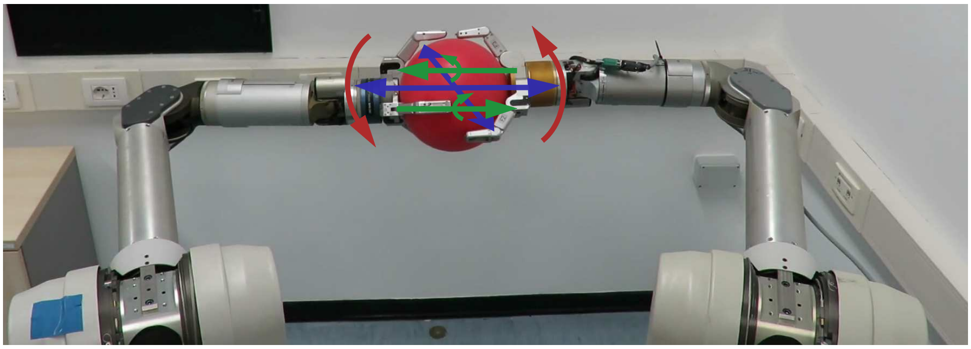


Figure 3.2: Visualization of the three different behaviors: i) horizontal planar translation (in blue), ii) vertical planar rotation (in red), iii) coupled rotation and translation (in green).

the desired LQR tracking precision can be related to the observed covariance, we formulate the cost function (3.1) in the tangent space $\mathcal{T}_{\bar{\mu}}\mathcal{M}_s$ using (3.3)

$$c = \frac{1}{2}\int \Big(\underbrace{\mathrm{Log}_{\bar{\mu}}(\bar{p})^\top}_{\xi^\top} Q\, \underbrace{\mathrm{Log}_{\bar{\mu}}(\bar{p})}_{\xi} + u^\top R u\Big)\,\mathrm{d}t, \tag{3.4}$$

$$\text{with} \quad \bar{\mu} = \begin{bmatrix} \mu \\ 0 \end{bmatrix}, \quad Q = \begin{bmatrix} \Sigma^{-1} & 0 \\ 0 & 0 \end{bmatrix},$$

and R the control cost matrix. With the dynamical system (3.2) and cost function (3.4), the optimal state feedback controller

$$u = L\mathfrak{s} \overset{(3.3)}{=} L\,\mathrm{Log}_{\bar{\mu}}(\bar{p}), \tag{3.5}$$

can be computed. Similarly to classical infinite horizon LQR, the gain matrix, $L = R^{-1}B^\top X$, is obtained by solving the algebraic Riccati equation

$$AX + AX - XBR^{-1}B^\top X + Q = 0. \tag{3.6}$$

Figure 3.1 demonstrates the approach on the manifold $\mathcal{S}^2 \times \mathbb{R}^2$. It shows how the response of the system changes based on the shape of the covariance matrix. LQR is computed in the tangent space of the attractor μ. Its response is visualized in the tangent space, and projected on the manifold.

3.4. Synergies in Bi-manual Manipulation

The presented method is tested in a bi-manual task. Our aim is to demonstrate that a variety of synergies can be learned and reproduced using our approach.

The experimental setup consists of two Barrett WAMs with three fingered hands. Together, the two end-effectors hold a ball. We evaluate three different synergies: i) horizontal planar translation; ii) vertical planar rotation; iii) translation coupled with rotation. The setup with an illustration of the coordination patterns is shown in Figure 3.2.

The synergies are taught through kinesthetic teaching [100]. For each synergy we demonstrated the tolerated motion of the hands around a desired ball pose. For example, behavior iii) is demonstrated by repeatedly moving both hands unidirectionally away from the center while rotating along the axis of motion. The demonstration data consist of hand-pose pairs which are defined at the hand palms.

The demonstration data of the bi-manual skill lie on the 12-dimensional ($d = 12$) manifold

$$\mathcal{M}_{bm} = \underbrace{\mathbb{R}^3 \times \mathcal{S}^3}_{\text{Pose 1}} \times \underbrace{\mathbb{R}^3 \times \mathcal{S}^3}_{\text{Pose 2}}. \tag{3.7}$$

For each behavior we computed the MLE of the mean and covariance of the Riemannian Gaussian $\mathcal{N}_{\mathcal{M}_{bm}}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and added $1{\cdot}10^{-3}\boldsymbol{I}_d$ as prior to the covariance. This regularization term prevents high gains in the state variables that have very low variance; it bounds the gains found through LQR.

The resulting models are visualized in Figure 3.3. The covariance of the position clearly shows the preferred direction of motion: i) motion in the horizontal plane; ii) motion in the vertical plane; iii) motion along one axis. Similarly, the rotational covariance provides information about the alloted rotation: i) no rotation in any direction; ii) rotation in a single plane; iii) rotation around a single axis.

The learned synergies appear in the correlation matrices[1]. For i), the planar coupling between the hand positions results in the strong positive correlation between $x_{L,1}$ and $x_{R,1}$, and between $x_{L,2}$ and $x_{R,2}$. For ii), the synergy involves a rotation around the global x_1 axis. This is correctly captured in the correlation between $\omega_{L,1}$ and $\omega_{R,2}$. Furthermore, there exists a strong negative correlation between the x_3 axes of the left and right hand. This indicates the opposite upwards/downwards motion made during the rotation. Note that the rotation of the hands around the ball created a circular motion around its center. This requires a nonlinear coupling between the x_2 and x_3 of both hands, something that cannot be properly captured in a single Gaussian. For iii), strong correlation is observed between $x_{2,L}$ and $x_{2,R}$, indicating the motion along the axis of translation. The strong correlation between $x_{2,L}$, $x_{2,R}$ and $\omega_{2,L}$, $\omega_{2,R}$ establishes the coupling between the translation and rotation.

To reproduce the demonstrated synergies, we employ the Riemannian LQR presented in Section 3.3. The system state manifold for the bi-manual skill

$$\mathcal{M}_{\boldsymbol{s}} = \underbrace{\mathbb{R}^3 \times \mathcal{S}^3}_{\text{Pose 1}} \times \underbrace{\mathbb{R}^3 \times \mathcal{S}^3}_{\text{Pose 2}} \times \underbrace{\mathbb{R}^3 \times \mathbb{R}^3}_{\text{Pose 1 velocity}} \times \underbrace{\mathbb{R}^3 \times \mathbb{R}^3}_{\text{Pose 2 velocity}}, \tag{3.8}$$

consists of the skill-manifold augmented with the pose velocities. We define a linear system in the tangent space $\mathcal{T}_{\tilde{\boldsymbol{\mu}}}\mathcal{M}_{\boldsymbol{s}}$ (3.2), where $\boldsymbol{M}$ and $\boldsymbol{C}$ are the end-effector inertia and

[1]The covariance matrix combines correlation coefficients $-1 \le \rho_{ij} \le 1$ among random variables X_i and X_j with deviation σ_i of random variables X_i, i.e. it has elements $\Sigma_{ij} = \rho_{ij}\sigma_i\sigma_j$. We prefer to visualize the correlation matrix instead of the covariance matrix, since it only contains the correlation coefficients and highlights the coordination among variables.

damping, which we approximate by

$$\begin{aligned} \boldsymbol{M} &= \mathrm{diag}(1.17, 1.17, 1.17, 0.009, 0.008, 0.005, \\ &\qquad 1.17, 1.17, 1.17, 0.009, 0.008, 0.005), \\ \boldsymbol{C} &= \mathrm{diag}(20, 20, 20, 1, 1, 0.1, 20, 20, 20, 1, 1, 0.1), \end{aligned}$$

respectively. We run the controller with a frequency of 500 Hz.

The control cost matrix $\boldsymbol{Q}$ is constructed from the inverse covariance matrix of the Gaussian describing the desired synergy

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{\Sigma}^{-1} & \boldsymbol{0}_{d\times d} \\ \boldsymbol{0}_{d\times d} & \boldsymbol{0}_{d\times d} \end{bmatrix}, \tag{3.9}$$

hereby setting a zero cost on the desired velocity. Furthermore, we manually defined a constant control cost matrix

$$\begin{aligned} \boldsymbol{R} = \mathrm{diag}(&\frac{1}{80}, \frac{1}{80}, \frac{1}{80}, 12.5, 12.5, 25, \\ &\frac{1}{80}, \frac{1}{80}, \frac{1}{80}, 12.5, 12.5, 25), \end{aligned}$$

which was the same for all three synergies.

By solving the LQR problem we obtain the gain matrix $\boldsymbol{L} \in \mathbb{R}^{12\times 24}$, which we use to compute the control command $\boldsymbol{u} = [\boldsymbol{u}_L^\top, \boldsymbol{u}_R^\top]^\top \in \mathbb{R}^{12}$. The desired joint torques are computed using

$$\boldsymbol{\tau} = \boldsymbol{\tau}_g(\boldsymbol{q}) + \boldsymbol{J}(\boldsymbol{q})^\top \boldsymbol{u}, \tag{3.10}$$

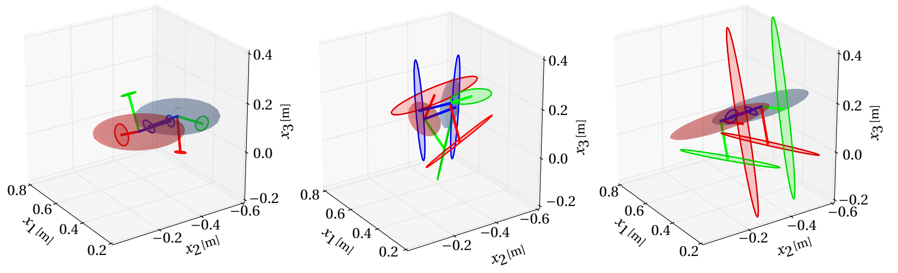
with $\boldsymbol{\tau}_g$ the torques required for gravity compensation, $\boldsymbol{J}$ the manipulator Jacobian, and $\boldsymbol{q}$ the joint angles.

Typical reproductions of the encoded synergies are visualized in Figure 3.3c and in the video accompanying this Chapter. Figure 3.4 shows the step response of the real system and the linear system (3.2) for behavior iii). The figure shows a stable response of the real system. The transients of the real system and the linear system are similar. The steady-state errors of the real system are likely due to the approximated inertia matrix and the unmodeled static friction.

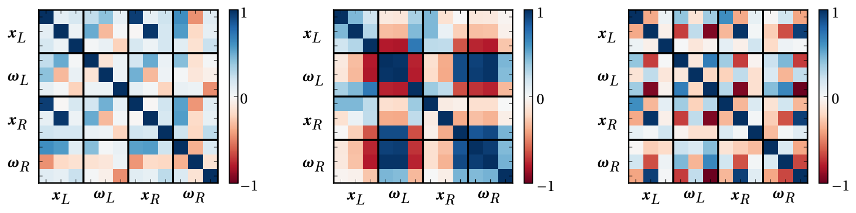
3.4.1. Quality of the synergies

Our approach exploits the structure of the manifold to discover the synergies. Yet, is the Riemannian approach better in extracting synergies than the Euclidean approach? In other words, would the same synergies emerge if we consider the quaternion data to be embedded in a 4D Euclidean space?

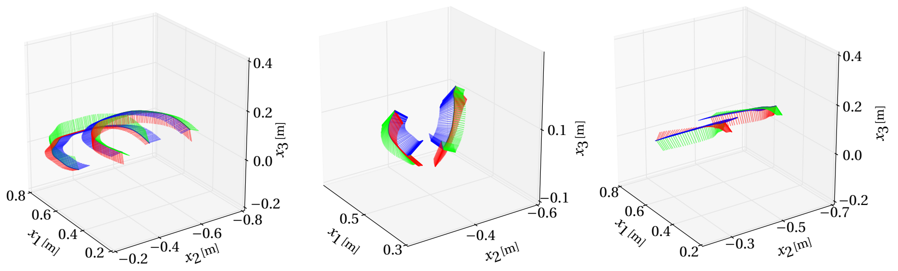
To assess this question, we analyze the spectral properties of the covariance matrices for the Riemannian Gaussian, and the 'Euclidean' Gaussian (where we treat Quaternion data as Euclidean). The Eigen vectors of the covariance matrix represents the synergies—functional couplings among the manifold dimension—and the Eigen values their importance (or strength).



(a) Graphs depicting the synergy models. The colored ellipsoids show the covariance of the left and right end-effectors in red and blue, respectively. The orthogonal red, green and blue lines indicate the orientation distribution. The colored ellipsoids at the end of each line depict the rotational covariance using $\sqrt{0.3}$-standard deviation. (see also the illustration guide on page xi).



(b) Correlation matrices. The entries related to the position and orientation of the left and right end-effectors are labeled $\boldsymbol{x}_{R/L}$ and $\boldsymbol{\omega}_{R/L}$, respectively.



(c) Reproduction examples. The orthogonal red, blue and green lines originate from the end-effector position and depict the end-effector orientation. The lines represent the end-effector x_1, x_2 and x_3 axes respectively.

Figure 3.3: Visualization of coordination encoding described in Sec. 3.4. The three behaviors (i)–(iii) are ordered left-to-right.

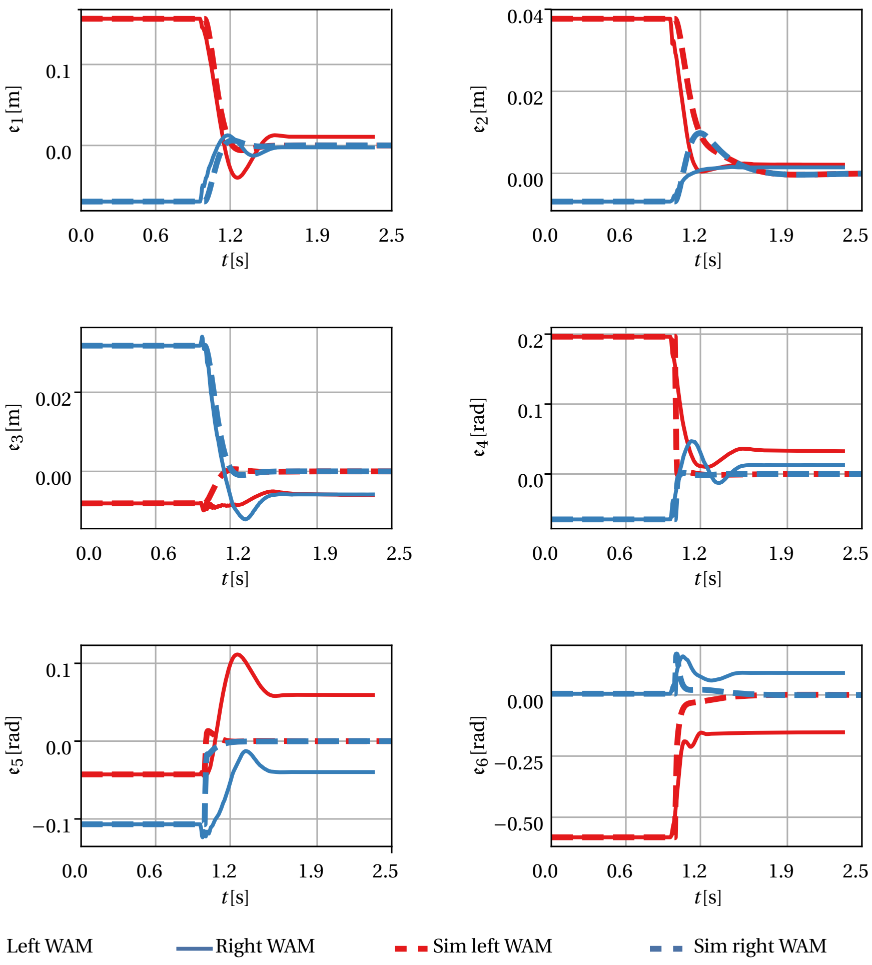
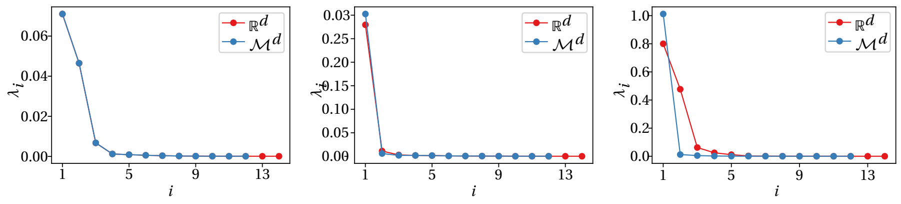


Figure 3.4: Step responses of the real and simulated linear system, visualized in the tangent space of the target state. The elements $\mathfrak{e}_1$–$\mathfrak{e}_3$, and $\mathfrak{e}_4$–$\mathfrak{e}_6$ correspond to the position and orientation, respectively.



(a) Translation (b) Planar rotation (c) Coupled Translation-Rotation

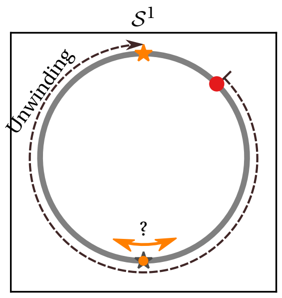
Figure 3.5: Visualization of the (sorted) Eigen values of the three different experiments presented in Section 3.4

From the three behaviors we demonstrated, behavior (i, Translation) should contains two degrees of freedom (the plane) and thus is expected to have two non-zero Eigenvalues. The planar rotation (behavior ii), requires a circular motion in the plane along the spatial degrees of freedom. As this behavior cannot be properly represented in asingle (linear) covariance matrix, more than one non-zero Eigen values can be expected when analyzing this motion pattern. Finally, the translation-rotational motion (iii) requires only a single mode of motion and should therefore only have one non-zero Eigen value.

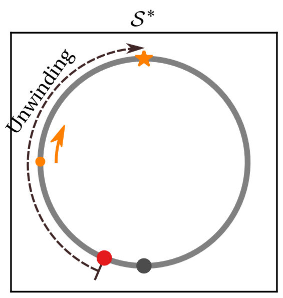
Figure 3.5 shows the Eigen values that are extracted from the three demonstrated synergies. The Riemannian approach contains 12 Eigenvalues (corresponding to the degrees of freedom in the system), and the Euclidean approach 14 (the additional 2 result from the redundant dimensions introduced by the Euclidean approximation of the quaternions). The translational behavior (a) displays similar results for the Riemannian and Euclidean approach: three non-zero Eigen values of which the first two correspond to the planar translation. The planar rotation clearly shows one large and one smaller non-zero Eigen value for both approaches. Finally, a clear difference between the Riemannian and the Euclidean approach is shown in the coupled translation/rotation. Here, the Riemannian approach correctly displays a single non-zero Eigen value, while the Euclidean incorrectly inferred 3. This latter result indicates that the Riemannian approach could be more suitable of encoding synergies when considering functional coupling along the translational and rotational degrees of freedom.

3.5. Discussion

This chapter presented an approach to learn task-space synergy controllers from demonstration data. We proposed to exploit Riemannian geometry to combine manifolds through the Cartesian product. This makes the approach easily adaptable to a variety of manifolds. For example, all experimental evaluations of LQR presented in this work are performed using one single piece of code. Changing from the toy-example to the bi-manual manipulation example solely required specifying a different system state manifold.



(a) Manifold $\mathcal{S}^1$



(b) Manifold double covering $\mathcal{S}^1$: $\mathcal{S}^*$

Figure 3.6: Visualization of the manifolds $\mathcal{S}^1$ and its double covering $\mathcal{S}^*$. See Section 3.5 for a detailed description.

The geometrical nature of the orientation group SO(3) prevents the existence of continuous globally stable state-feedback controllers [104, 105]. We illustrate the geometric nature of this problem using Figure 3.6. In this figure the unit sphere $\mathcal{S}^1$ and its double covering $\mathcal{S}^*$ are considered as an illustrative analogy to SO(3) and its double covering $\mathcal{S}^3$. First, we describe why a state-feedback controller on SO(3) is discontinuous and not globally stable. Then, we explain why a double covering of the rotational group does not resolve the issue.

The discontinuity in control on SO(3) arises when the angle error between the setpoint and system state has a magnitude of $\pm\pi$, as illustrated in Figure 3.6a by the orange dot (system state) and stars (setpoint). In this scenario, there is no unique minimizing path between the setpoint and the system state; the system state lies at the antipodal of the setpoint. The control signal of a state-feedback controller switches sign when passing the antipodal, and therefore is discontinuous. In addition, the state-feedback controller is not globally stable, because the antipodal is an (unstable) equilibrium.

$\mathcal{S}^*$ double covers $\mathcal{S}^1$; making one rotation about $\mathcal{S}^*$ yields two rotations on $\mathcal{S}^1$. By controlling orientation on this double covering, the maximum rotation angle is represented by a distance of only $\pi/2$. The discontinuity that appeared on $\mathcal{S}^1$ seems to be resolved: at any point in the top hemisphere—which fully covers $\mathcal{S}^1$—there is only one shortest path to the orange setpoint. However, this solution does not really resolve the original problem: a discontinuity still exists when the system state is at the antipodal of $\mathcal{S}^*$ (which is the setpoint itself in $\mathcal{S}^1$). In addition, the solution gives rise to the 'unwinding' phenomena [104]: when the state is close to the antipodal of $\mathcal{S}^*$ it moves towards the opaque orange setpoint, thereby unwinding the double covering. As a result, the system state will first move away from the setpoint before converging to it.

In our definition of the logarithmic map (see Table 2.1), we ensure that we always measure the minimum distance between two quaternions. This avoids the unwinding phenomena, but maintains the discontinuity at $\pm\pi$ state error and absence of global stability. However, the attraction domain of the unstable equilibria is nowhere dense [105]. Furthermore, by definition of our LQR problem, this unstable equilibrium lies π radian away from the setpoint ($\boldsymbol{\mu}$). Therefore, we consider the absence of global convergence a theoretical limitation that has no practical consequence to the learning and reproduction of synergies from demonstration.

The presented approach relies on the ability to measure minimum distances on the manifold. This is not naturally achieved on $SE(3)$ because neither a bi-invariant metric nor a natural left or right invariant metric exists [86]. In this work, we choose to use a left-invariant metric because it allows us to encode the synergy models independent from the inertial frame.

The state-feedback controller (3.5) is similar to the double-geodesic controller for Lie-groups presented by Bullo and Murray [106]. In their work on PD-control, they also highlight the difference between control on SE(3) and SO(3) $\times \mathbb{R}^3$ (or $\mathcal{S}^3 \times \mathbb{R}^3$, as we do). They show that controllers defined on SE(3) do not follow geodesics of the SE(3) manifold. But controllers defined on the Cartesian product SO(3) $\times \mathbb{R}^3$ do. Yet, when controlling on $\mathcal{S}^3 \times \mathbb{R}^3$ we still need to specify a relative weighting between positional and rotational components. Since a similar trade-off is faced when balancing torques and forces through $\boldsymbol{R}$, we choose to equally weight position and orientation contributions in the metric. The proper weighting of position and orientation is then postponed until the inevitable tuning of the control cost matrix.

The control cost matrix remains an open parameter that requires manual tuning. In our experimental evaluation, we set the values of $\boldsymbol{R}$ in such a way that the control effort is well balanced for the translation and rotational degrees of freedom. The fact that we could use a single $\boldsymbol{R}$ for the three different synergies shows that its selection is more system than task dependent. In practice, one could specify a fixed system-dependent ratio $\boldsymbol{R}_f$ for the control variables and allow the user to control the overall control cost of the system by a single parameter β, i.e. $\boldsymbol{R} = \beta\boldsymbol{R}_f$. Here, $\boldsymbol{R}$ should be bounded to prevent unstable controllers due to actuator limitations of the real systems.

Throughout this chapter, we focused on the derivation of a controller for a single synergy. Real manipulation tasks will require a variety of synergies along a desired trajectory. By encoding manipulation tasks in a Gaussian Mixture Model (GMM), representing the joint distribution of time and pose $\mathcal{P}\left(t, \boldsymbol{p}\right)$, one can for example obtain a time dependent synergy by computing $\mathcal{P}\left(\boldsymbol{p}|t\right)$ using Gaussian Mixture Regression (GMR) (see Section 2.5.7). In situations where regression is computationally demanding, one could run the regression at a relatively low rate, while the presented controller ensures a synergetic response to disturbance.

4 Context-Adaptive Tasks

4.1. Introduction

Generalization is a key requirement in Programming by Demonstration (PbD) [2,3], and robot learning in general. It is the ability to model, synthesize and recognize tasks performed in different contexts. In this chapter, we focus on generalization through context independent task modeling.

Before elaborating, we define the terms *task context* and *context independence*. Consider sending a parcel. This involves filling a box with items to ship, closing it and attaching an address label. Parcels differ in shape and size, and might be packed at different position and orientation. This information is considered as *task context*: the circumstances under which a task is executed. The motions required to pack, close and mark a parcel depend on the task context. Each task has characterizing features: the address label in the center, the stamp in the upper right corner, etc. These features are intrinsic to the task. A model that represents these features free of context, is *context independent*.

Task modeling, synthesis and recognition are three common actions for a robotic system: modeling allows robots to maintain compact task representations and improve them over time; through synthesis robots generate motion, allowing them to act and change the state of the world; and task recognition enhances situation awareness and communication capabilities, as it enables the robot to identify tasks or gestures.

A robot that models skills independent of context, can adapt tasks more effectively to unseen context. Instead of maintaining separate task models for each context, every task only requires one context-independent model. This makes PbD more effective, as demonstrations of a task observed in different context can train a single context-independent model. This reduces the number of demonstrations required, and removes the need to demonstrate the task in a particular context. The smaller set of task models potentially improves recognition accuracy, as there are less classes to distinguish. Additionally, task recognition could be augmented with context inference—the ability to estimate in which context a task was executed. Finally, context independent models improve synthesis, which is achieved by associating new context with the context independent model.

This chapter presents a context independent framework for task space manipulation based on the task-parameterized framework [18, 21, 40]. This framework achieves strong generalization by relying on generic, but pre-defined, model and context structures. The contributions of this chapter with respect to previous works are threefold:

1. it demonstrates how the task-parameterized framework is generalized to Riemannian manifolds using the material presented in Chapter 2;
2. it provides an overview of the type of context parameterizations that are applicable in the (Riemannian) task-parameterized framework;
3. it completes the task-parameterized framework with an algorithm to infer task context, next to the already existing methods for modeling and synthesis.

After discussing the related work in Section 4.2, Section 4.3 reviews the task-parameterized framework and introduces the extension to Riemannian manifolds and context inference. Then, the modeling, synthesis and inference actions are evaluated in Section 4.4, and more practical applications are presented in Section 4.5.

4.2. Related Work

Context adaptation is related to two elementary questions in the field of PbD, namely: *What to imitate?* (what features represent the task) and *How to imitate?* (how to adapt them to different context) [2]. Commonly, context-adaptive approaches answer the former using demonstration data, and the latter by pre-defined model and context structures.

The Dynamic Movement Primitives (DMP) framework [8, 24] has different context-adaptive implementations. A DMP is a linear spring-damper system that is perturbed by a non-linear forcing term. As the non-linear term decays, the system is guaranteed to converge to the spring attractor (goal position). DMP thus allows the modification of both the start and goal position. This can be seen as a form of context adaptation, and has been used to generalize end-effector motions in task-space [25, 27, 28, 107]. Pervez and Lee [31] propose a task-parameterized DMP. Here, the context parameters are modeled in a joint probability density function together with the DMP parameters. Doing so, the DMP can adapt to via-point context in addition to the start and goal contexts. Their experimental results demonstrate both interpolation and extrapolation capabilities. Yet, the examples are restricted to $2D$ planar motion with spatial context only. Ude *et al.* [26] proposed a variant of DMP that encodes the full end-effector pose. This extension allowed DMP to adapt to changes in both goal position and orientation. It has been used in several task space applications which required context adaptation, e.g. [29, 30]. Although DMP can adapt to changing goals, its lack of spatial coordination precludes the use of spatial constraints to reach them (e.g. when seizing a book from a shelf under a constrained motion direction). Furthermore, it remains unclear how DMP can handle more expressive context, which comprises multiple objects or object scaling. The context-adaptive approach presented in this chapter is able to take such situations into account.

The more recently proposed Probabilistic Movement Primitives (ProMP) [22] has several context-adaptive extensions. As DMP, it supports the modification of start and goal position. Additionally, as ProMP encodes spatial coordination, it allows for constrained reaching. A more elaborate coupling between task context and task motion can be achieved by encoding a joint distribution of context and ProMP weights. During task synthesis context dependent weights are obtained through conditioning [23]. A similar approach can be used to coordinate movements between human and robot [108]. Yet, these approaches only consider position-based context, and rely on linear regression which most often remain feasible for interpolation within known contexts only.

The context parameterization presented by Brandi *et al.* [109] seems similar to the one presented in this chapter. Brandi *et al.* relate the context to coordinate systems (or landmarks) defined on template objects. The context parameters are obtained from a point-cloud matching algorithm that finds the warped transformation between a known point-cloud and an observed object. The context is parameterized by position and a number of (Euler) angles. These context parameters are used to define a context independent ProMP. While the method presented by Brandi *et al.* only allows the transformation of the ProMP mean, our approach provides a variety of transformations (including translation, scaling and rotation), and applies these to the full model (mean and covariance).

The task-parameterized (TP) approach [18, 21, 40] models the demonstration in different coordinate systems (frames of reference). These coordinate systems are linked to the position or orientation of different objects and landmarks in the environment—the task context. Examples are: the position and orientation of table legs in an assembly task [100], location of a box in pointing and pick-up tasks [19, 40], or parts of the robots body [40, 41]. Based on the demonstration data, each coordinate system encodes a unique model of the task motion. By demonstrating the task for varying context, unique motion patterns appear in each coordinate systems. Invariance in these patterns corresponds to the importance of the coordinate systems. During motion synthesis invariance information is used to fuse the information encoded in the different coordinate systems. This fusing step can be seen as maximization of a metric of imitation. Such a metric has been used to generalize demonstration data in [33, 110]. In a similar fashion, Bowen *et al.* [111] and Yang *et al.* [112] provide ways to learn imitation metrics for Euclidean data. The use of multiple coordinate systems in combination with a fusion method provides strong generalization capabilities. Even though the approach has originally been presented using GMM-based skill representations, it is also compatible with representations such as ProMP or Gaussian Process (GP) [21].

Methods based on invariant features represent a distinct branch of movement encoding [113–115]. These methods represent task-space motion using velocity-like descriptors and can encode both position and orientation information. The removal of explicit spatial and temporal information creates context-independent task representations. Vochten *et al.* [114] combine invariant features with optimization to generalize demonstrated point-to-point trajectories to new start and goal poses. Lee *et al.* [115] use a feature representation that is bi-directional; its transformation between the Cartesian space and invariant space is achieved without losing information. This approach is appealing and similar to our context-adaptive framework in the sense that it is able to generalize, encode, synthesize and recognize motion in varying context. Similar to our framework, the context adaptation can include affine transformations. Yet, unlike Lee *et* *al.*, the approach presented in this Chapter can take multiple task-relevant objects into account.

Alternatively, exploration-based methods have been considered for context adaptation. Examples of such methods are reinforcement learning [10,116,117], and deep reinforcement learning [118, 119]. Despite appealing, these methods typically require large amount of real-world interaction, which are not always available in our applications.

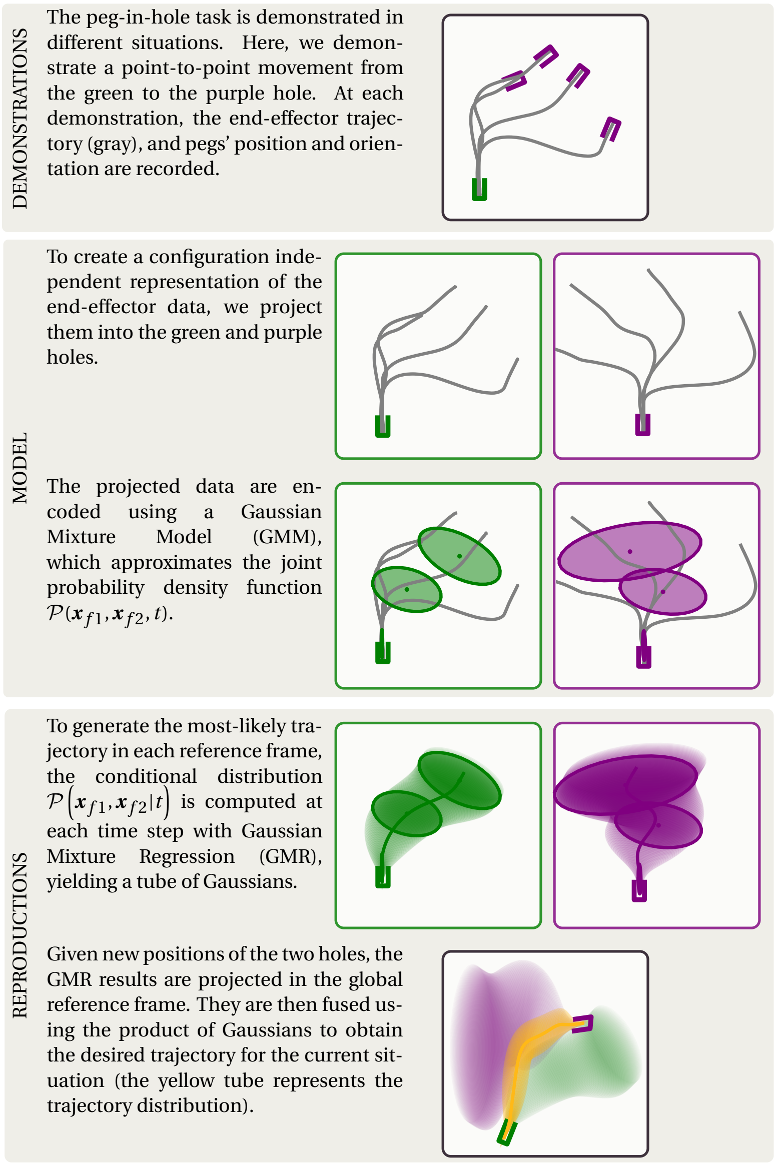
4.3. RIEMANNIAN TASK-PARAMETERIZED GAUSSIAN MIXTURE MODEL

The task-parameterized framework presented in this chapter achieves strong context adaptation by representing task motion in local coordinate systems. It builds on the Task-Parameterized Gaussian Mixture Model (TP-GMM) presented by Calinon [21]. The framework requires three elements: a context parameterization which defines the relation between local (context-independent) coordinate systems, and a global coordinate system; a model parameterization to compactly represent the local movement representations; and a merging method that allows to fuse the contextualized movement representations. Before discussing these elements in detail, we give a conceptual description of the method using Figure 4.1.

The task-parameterized framework observes demonstrations from different perspectives. Typically, these perspectives are linked to the task context—objects or landmarks in the environment. In Figure 4.1, the perspectives are the green and purple holes. To formalize the context, it is represented by coordinate systems; one coordinate system attached to each object or landmark. The task parameters describe the relation between the local coordinate systems and the global coordinate system. In Figure 4.1 this relation is a rigid-body transformation, but this chapter demonstrates more complex relations are possible.

By observing demonstrations from different perspectives, variant and invariant regions appear. Invariance indicates that the state of an object—which is linked to the coordinate system—influences the task motion; it indicates that an object state is important for task execution. This becomes apparent in Figure 4.1, where invariance appears close to the green and purple holes. We call these local representations context independent, because they represent the task isolated from a specific context. In order to compactly represent the local task representations without losing the covariance information, it is encoded in a joint probability density function (pdf). In this chapter, we represent this pdf using a (Riemannian) Gaussian Mixture Model (GMM), the TP-GMM.

The encoding of local variability enables the generalization to new contexts. To reproduce the encoded behavior, the local representations are placed in a global coordinate system according to the newly encountered context. There, the local representations are fused while taking into account the encoded variability. Figure 4.1 shows how Gaussian Mixture Regression (GMR) and product of Gaussians are used to generalize the peg-in-hole task to a new context. Note that generalization takes into account both the position and orientation of the holes. Doing so, the generalization respects the constrains required to enter the holes.



The peg-in-hole task is demonstrated in different situations. Here, we demonstrate a point-to-point movement from the green to the purple hole. At each demonstration, the end-effector trajectory (gray), and pegs' position and orientation are recorded.

To create a configuration independent representation of the end-effector data, we project them into the green and purple holes.

The projected data are encoded using a Gaussian Mixture Model (GMM), which approximates the joint probability density function $\mathcal{P}(\boldsymbol{x}_{f1}, \boldsymbol{x}_{f2}, t)$.

To generate the most-likely trajectory in each reference frame, the conditional distribution $\mathcal{P}\left(\boldsymbol{x}_{f1}, \boldsymbol{x}_{f2} | t\right)$ is computed at each time step with Gaussian Mixture Regression (GMR), yielding a tube of Gaussians.

Given new positions of the two holes, the GMR results are projected in the global reference frame. They are then fused using the product of Gaussians to obtain the desired trajectory for the current situation (the yellow tube represents the trajectory distribution).

Figure 4.1: Conceptual description and illustration of TP-GMM. A further analysis of this example is given in Section 4.4.2.

Formally, a TP-GMM represents the probability distribution

$$\mathcal{P}\left(\boldsymbol{x} \,|\, \boldsymbol{\theta}^{\mathrm{m}}, \boldsymbol{\theta}^{\mathrm{c}}\right) = \sum_{k=1}^{K} \pi_k \prod_{p=1}^{P} \mathcal{N}\left(\boldsymbol{x} \,|\, \boldsymbol{\theta}^{\mathrm{m}}_{\mathrm{p,k}}, \boldsymbol{\theta}^{\mathrm{c}}_{\mathrm{p}}\right). \tag{4.1}$$

It consists of a weighted sum of K distributions, with the weights $\sum_{k=1}^{K} \pi_k = 1$ and $\pi_k \geq 0$.

The summed distributions are formed by a product of P Gaussians $\mathcal{N}\left(\boldsymbol{x}\,|\,\boldsymbol{\theta}^{\mathrm{m}}_{\mathrm{p,k}}, \boldsymbol{\theta}^{\mathrm{c}}_{\mathrm{p}}\right)$; each representing the task in a local coordinate system. The Gaussians are defined by the model parameters $\boldsymbol{\theta}^{m}_{p,k} = \{\boldsymbol{\mu}_{p,k}, \boldsymbol{\Sigma}_{p,k}\}$, and context parameters $\boldsymbol{\theta}^{c} = \{\boldsymbol{\theta}^{c}_{p}\}^{P}_{p=1}$.

The task-parameterized framework involves three elements: (contextualized) demonstration data, task context, and a context-independent model. The elements form a triad; given two, one can estimate the other. This triad leads to three optimization problems, namely model estimation, motion synthesis and context inference. The objectives of these optimization problems are listed in Table 4.1.

The concepts for model estimation and motion synthesis have been discussed in previous work [21] and will be briefly reviewed in sections 4.3.2 and 4.3.3. Context inference has not yet been addressed in the light of TP-GMM. Section 4.3.4 describes the inference problem, and proposes an Expectation Maximization (EM)-based solution. With the introduction of context inference, one can implement task recognition (the last entry in Table 4.1) straightforwardly. This allows the recognition of tasks from pre-defined library of TP-GMM. Yet, before we can describe modeling, synthesize and inference, the task parameters, and their relation to the model parameters needs to be defined. This is done in Section 4.3.1.

Action	Optimization objective	
Modeling	$\boldsymbol{\theta}^{\mathrm{m}}\left(\boldsymbol{x}_{1:N}, \boldsymbol{\theta}^{\mathrm{c}}_{1:N}\right)$	$= \arg\max_{\boldsymbol{\theta}^{\mathrm{m}}} \prod_{n=1}^{N} \mathcal{P}\left(\boldsymbol{x}_n\,|\,\boldsymbol{\theta}^{\mathrm{m}}, \boldsymbol{\theta}^{\mathrm{c}}_{n}\right)$
Synthesis	$\boldsymbol{x}^{\mathcal{O}}\left(\boldsymbol{x}^{\mathcal{I}}, \boldsymbol{\theta}^{\mathrm{m}}, \boldsymbol{\theta}^{\mathrm{c}}\right)$	$= \arg\max_{\boldsymbol{x}^{\mathcal{O}}} \mathcal{P}\left(\boldsymbol{x}^{\mathcal{O}}\,|\,\boldsymbol{x}^{\mathcal{I}}, \boldsymbol{\theta}^{\mathrm{m}}, \boldsymbol{\theta}^{\mathrm{c}}\right)$
Inference	$\boldsymbol{\theta}^{\mathrm{c}}\left(\boldsymbol{x}_{1:N}, \boldsymbol{\theta}^{\mathrm{m}}\right)$	$= \arg\max_{\boldsymbol{\theta}^{\mathrm{c}}} \mathcal{P}\left(\boldsymbol{x}_{1:N}\,|\,\boldsymbol{\theta}^{\mathrm{m}}, \boldsymbol{\theta}^{\mathrm{c}}\right)$
Recognition	$t\left(\boldsymbol{x}_{1:N}, \boldsymbol{\theta}^{\mathrm{m}}_{1:T}, \boldsymbol{\theta}^{\mathrm{c}}_{1:T}\right)$	$= \arg\max_{t\in\{1,\ldots,T\}} \mathcal{P}\left(\boldsymbol{x}_{1:N}\,|\,\boldsymbol{\theta}^{\mathrm{m}}_{t}, \boldsymbol{\theta}^{\mathrm{c}}_{t}\right)$

Table 4.1: Overview of optimization objectives in the task-parameterized framework. The symbols $\boldsymbol{\theta}^{\mathrm{c}}$ and $\boldsymbol{\theta}^{\mathrm{m}}$ refer to the context and model parameters respectively. $a : b$ indicates a set of elements, e.g. $\boldsymbol{\theta}^{c}_{1:N} = \{\boldsymbol{\theta}^{c}_{1}, \cdots, \boldsymbol{\theta}^{c}_{N}\}$. The superscripts $\mathcal{I}$ and $\mathcal{O}$ indicate input and output, respectively.

In the task-parameterized framework, the task is considered from the landmarks and objects that form the task context. The motion is thus described in the coordinate systems that are attached to objects and landmarks. Here, we call these local representations context independent, as they describe the robot motion independently from a context instance (i.e. a specific location, or shape).

When demonstrating the task for varying context, different invariant patterns will emerge in these coordinate systems. The variability among the demonstrations observed in each coordinate system indicates its importance in the task. When variability is large, the robot motion was not consistent with the object's pose or shape. In this case the object's state is not important for the task motion. For low variability the opposite holds. It is important to realize that this importance can be time-varying; the object state can be important at the start of the motion and not at the end of the motion. After projecting the data into the local coordinate systems, they can be encoded in a compact probabilistic model. To synthesize the data in a new context, the local models are first contextualized—i.e. projected into a common (global) coordinate system. There, the contextualized models need to be fused to obtain the task motion for the current context. The fusion involves a trade-off in which each coordinate system is weighted by the observed variance.

4.3.1. Context Parameterization

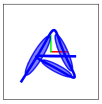
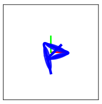
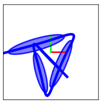
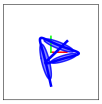
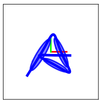
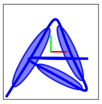
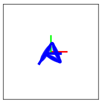
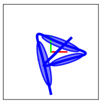
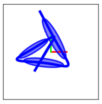
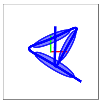
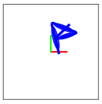
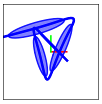
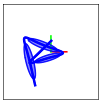
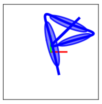
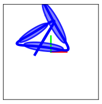
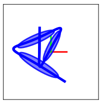
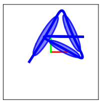
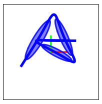
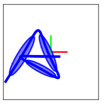
Group Name	Group Definition	Template Deformations		
Template				
General Linear Group	$\mathrm{GL}(n)$			
Uniform Scale Group (USG)	$\mathrm{US}(n) = \{\boldsymbol{Q} : SI_{n\times n}, S \in \mathbb{R}^+\}$			
Special Orthogonal Group	$\mathrm{SO(n)} = \{\boldsymbol{Q} : \boldsymbol{Q}\boldsymbol{Q}^\top = I, \det(\boldsymbol{Q}) = 1\}$			
Affine Group	$\mathrm{Aff(n)} = \left\{ \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{v} \\ \boldsymbol{0} & 1 \end{bmatrix} : \boldsymbol{Q} \in \mathrm{GL}(n), \boldsymbol{v} \in \mathbb{R}^n \right\}$			
Special Euclidean Group	$\mathrm{SE(n)} = \left\{ \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{v} \\ \boldsymbol{0} & 1 \end{bmatrix} : \boldsymbol{Q} \in \mathrm{SO(n)}, \boldsymbol{v} \in \mathbb{R}^n \right\}$			
Translational Group	$\mathrm{TG(n)} = \left\{ \begin{bmatrix} \boldsymbol{I}_{n\times n} & \boldsymbol{v} \\ \boldsymbol{0} & 1 \end{bmatrix} : \boldsymbol{v} \in \mathbb{R}^n \right\}$			

Table 4.2: An overview of the groups that are considered for context parameterization. To illustrate the difference between the groups, we apply three different transformations to a template shape (visualized in the second row of the table). The red/green axis indicates the origin of the original template, and assists the visualization of rotation and translation.

TP-GMM relies on a context parameterization. It is defined by the task parameters, and their relation to the GMM model parameters. Originally, the linear transformation properties of the Gaussian represent this relation. They permit affine transformation of GMM parameters, and thus form an affine context parameterization. Similarly, transformation properties of the Riemannian Gaussian, defined in Section 2.5.5, are compatible with the affine group.

The transformation capacity of this group is abundant for task space manipulation, as it comprises any combination of scaling, translating and rotating of task-space motion. Table 4.2 gives an overview of these abilities for the affine group and its subgroups. There, the group operations are applied to a mixture of Euclidean Gaussians. The context parameterization $\{\boldsymbol{Q} \in \mathrm{GL}(3), \boldsymbol{v} \in \mathbb{R}^3\}$ allows translation, scaling and rotation of the original model.

Although the affine group is suitable for Euclidean data, it is not directly applicable to rigid-body poses. The application of an affine transformation to a rigid-body pose does not yield a rigid-body pose, but an affine transformation. This is exemplified for SE(3). Consider the affine transformation to a pose, namely

$$\underbrace{\begin{bmatrix} \boldsymbol{QR} & \boldsymbol{Qg}+\boldsymbol{v} \\ \boldsymbol{0} & 1 \end{bmatrix}}_{\mathrm{Aff}(3)} = \underbrace{\begin{bmatrix} \boldsymbol{Q} & \boldsymbol{v} \\ \boldsymbol{0} & 1 \end{bmatrix}}_{\mathrm{Aff}(3)} \underbrace{\begin{bmatrix} \boldsymbol{R} & \boldsymbol{g} \\ \boldsymbol{0} & 1 \end{bmatrix}}_{\mathrm{SE}(3)}. \tag{4.2}$$

It preserves the Euclidean structure of the pose its spatial part (i.e. $\boldsymbol{g}, \boldsymbol{Qg}+\boldsymbol{v} \in \mathbb{R}^3$), but does not necessarily preserves the group structure of its orientation part. As $\boldsymbol{Q} \in \mathrm{GL}(3)$ and $\boldsymbol{R} \in \mathrm{SO}(3) \subset \mathrm{GL}(3)$, the operation $\boldsymbol{QR}$ yields an element of GL(3). Only when $\boldsymbol{Q} \in \mathrm{SO}(3)$, $\boldsymbol{QR}$ is an element of SO(3), and (4.2) yields a valid pose.

In order to include orientation data into a TP-GMM, it needs to permit affine transformations. This is achieved by applying the affine transformation separately on the spatial and orientation parts of the rigid-body pose. The affine transformation is straightforwardly applied to the spatial part, and employs its desirable properties of rotation, translation and scaling. In addition, the orientation part $\boldsymbol{R}$ is transformed using the rotation $\boldsymbol{R}_Q \in \mathrm{SO}(3)$ which is closest to transformation $\boldsymbol{Q}$, namely

$$\boldsymbol{R}_{\boldsymbol{Q}} = \arg \min_{\boldsymbol{R} \in \mathrm{SO}(3)} \|\boldsymbol{QR}^\top - \boldsymbol{I}\|_F, \tag{4.3}$$

with $\|\cdot\|_F$ the Frobenius norm. The measure of closeness is based on the orthogonality property $\boldsymbol{RR}^\top = \boldsymbol{I}$; the matrix $\boldsymbol{R}$ which makes $\boldsymbol{QR}^T$ closest to the identity is defined to be the closest rotation. As we rely on the quaternion representation for orientation, we also define

$$\Psi_{\mathcal{S}^3} : \mathrm{GL}(3) \to \mathcal{S}^3, \tag{4.4}$$

which maps $\boldsymbol{Q} \in \mathrm{GL}(3)$ to the closest quaternion in the sense of (4.3).

After introduction of the affine transformations, we review its relation to the model parameters and define the task parameters. Recall, the transformation properties presented in Section 2.5.5. These properties can be used to transform a GMM with K Gaussians and base $\boldsymbol{e} \in \mathcal{M}$ using the task parameters $\boldsymbol{\theta}^{\boldsymbol{c}} = \{\boldsymbol{A}, \boldsymbol{b}\}$. The parameters of the contextualized Gaussian are thus defined by

	$\mathbb{R}^3$	$\mathcal{S}^3$
$\boldsymbol{A}$	$\boldsymbol{Q}$,	$\boldsymbol{I}_3$,
$\boldsymbol{b}$	$\boldsymbol{v}$,	$\Psi_{\mathcal{S}^3}(\boldsymbol{Q})$

Table 4.3: $\boldsymbol{A}$ and $\boldsymbol{b}$ parameterization for task-space manifolds.

$$\boldsymbol{\mu}_k^{\boldsymbol{A},\boldsymbol{b}} = \Psi_{\boldsymbol{\theta}^c}(\boldsymbol{\mu}_k) = \mathrm{Exp}_{\boldsymbol{b}}\big(\boldsymbol{A}\,\mathrm{Log}_{\boldsymbol{e}}(\boldsymbol{\mu}_k)\big), \tag{4.5}$$

$$\boldsymbol{\Sigma}_k^{\boldsymbol{A},\boldsymbol{b}} = \Phi_{\boldsymbol{\theta}^c}(\boldsymbol{\Sigma}_k) = \big(\boldsymbol{A}\boldsymbol{\Sigma}_k\boldsymbol{A}^\top\big)_{\|_{\boldsymbol{b}}^{\boldsymbol{\mu}_k^{\boldsymbol{A},\boldsymbol{b}}}}, \tag{4.6}$$

where the subscript $\|_{\boldsymbol{b}}^{\boldsymbol{\mu}_k^{\boldsymbol{A},\boldsymbol{b}}}$ refers to the parallel transport of a covariance matrix from $\boldsymbol{\mu}_k$ to the contextualized $\boldsymbol{\mu}_k^{\boldsymbol{A},\boldsymbol{b}}$, as described in Section 2.5.5. Table 4.3 indicates how an affine transformation, characterized by $\boldsymbol{Q} \in \mathrm{GL}(3)$ and $\boldsymbol{v} \in \mathbb{R}^3$, is reflected in task parameters of these manifolds. For the Cartesian product $\mathcal{M} = \mathbb{R}^3 \times \mathcal{S}^3$, their contributions are combined as,

$$\boldsymbol{A}_{\mathcal{M}} = \begin{bmatrix} \boldsymbol{A}_{\mathbb{R}^3} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_{\mathcal{S}^3} \end{bmatrix} \in \mathcal{T}_{\boldsymbol{e}}\mathcal{M}, \tag{4.7}$$

$$\boldsymbol{b}_{\mathcal{M}} = \left(\boldsymbol{b}_{\mathbb{R}^3}, \boldsymbol{b}_{\mathcal{S}^3}\right) \in \mathcal{M}. \tag{4.8}$$

To train the context-adaptive tasks through demonstration, maps are required that transform data between the context-dependent and context-independent representation. For this we define $\Psi_{\boldsymbol{\theta}^c}(\boldsymbol{x})$ and its inverse,

$$\Psi_{\boldsymbol{\theta}^c}(\boldsymbol{x}) = \mathrm{Exp}_{\boldsymbol{b}}\!\left(\boldsymbol{A}\,\mathrm{Log}_{\boldsymbol{e}}(\boldsymbol{x})\right), \tag{4.9}$$

$$\Psi^{-1}_{\boldsymbol{\theta}^c}(\boldsymbol{x}) = \mathrm{Exp}_{\boldsymbol{e}}\!\left(\boldsymbol{A}^{-1}\,\mathrm{Log}_{\boldsymbol{b}}(\boldsymbol{x})\right). \tag{4.10}$$

Note that transformation (4.5) equals (4.9).

4.3.2. Modeling

Task modeling is the process of estimating the context-independent model parameters $\boldsymbol{\theta}^{\mathrm{m}}$ from task examples performed under a given context. The input of this process are therefore the task context and task motion. Together, they form the demonstration data $\boldsymbol{X} = \{\boldsymbol{x}_n, \boldsymbol{\theta}^c_n\}_{1:N}$, with N the number of samples.

Algorithm 4.1 describes the modeling process. First, it projects each sample $\boldsymbol{x}_n$ into the P coordinate systems, and collects the projections in $\boldsymbol{X}_n$ (lines 2–8). This collecting ensures that the P projections of $\boldsymbol{x}_n$ are perceived as one data point during parameter estimation. The group of all projected samples forms the context-independent data set $\boldsymbol{X}^m$. If the demonstration data $\boldsymbol{x}_n$ lie on the manifold $\mathcal{M}$, the elements $\boldsymbol{X}^m$ lie on the manifold $\mathcal{M} \times \cdots \times_{P-1} \mathcal{M}$ (the Cartesian product of P task manifolds $\mathcal{M}$).

The context-independent data are modeled into a joint distribution represented using a GMM with K Riemannian Gaussians (line 11). Its parameters are estimated using EM as described in Section 2.5.6. The parameters of the distributions in each of the P coordinate systems are obtained by marginalization (line 15).

Algorithm 4.1 differs slightly from the TP-GMM formulation in [18,21]. These formulations use a modified EM-algorithm to estimate the parameters of P GMMs. During its E-step, the responsibilities (see also Section A.2), are computed using the joint distribution $\prod_p \mathcal{N}\left(\mathrm{x_n}, \boldsymbol{\theta}^{\mathrm{m}}_{\mathrm{p}}, \boldsymbol{\theta}^{\mathrm{c}}\right)$. This ensures that the likelihood of $\boldsymbol{x}_n$ is jointly evaluated on the P coordinate systems. These responsibilities are then used in the M-step, to compute the parameters of the P GMMs separately. In contrast, Algorithm 4.1 models the joint distribution over the context-independent data (line 11); it computes the correlation among the P coordinate systems in the M-step, and uses this in the E-step to compute the responsibilities. Essentially, this approach implies that both the variables among the coordinate systems, and variables observed within a coordinate system might be correlated.

As each locally projected data point originates from the same global data point, the correlation among coordinate systems is related to context correlation.

Practically, the modified formulation removes the need for a separate EM-algorithm for context-adaptive movements. In addition, the correlation information might be used to detect redundancy in context. Objects or landmarks are considered to be redundant if they affect task execution in a similar fashion. Context redundancy can be related to correlation: demonstration data projected in coordinate systems of redundant objects will have similar patterns, and are therefore correlated.

Algorithm 4.1 Pseudo-code of the task-adaptive model-estimation.

```
1: function MODELCONTEXTDATA($\{\boldsymbol{x}_n, \boldsymbol{\theta}^c_n\}_{1:N}, K$)
2:     # Create context independent data set
3:     $\boldsymbol{X}^m = \{\}$
4:     for $n \in \{1 \cdots N\}$ do
5:         $\boldsymbol{X}_n = \{\}$
6:         for $p \in \{1, \dots, P\}$ do
7:             AddToSet$\left(\Psi^{-1}_{\boldsymbol{\theta}^c_{n,p}}(\boldsymbol{x}_n), \boldsymbol{X}_n\right)$        ▷ Using (4.10)
8:         AddToSet$\left(\boldsymbol{X}_n, \boldsymbol{X}^{\mathrm{m}}\right)$
9:
10:    # Model a joint distribution over multiple coordinate systems
11:    $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k\}_{k=1}^K \leftarrow \mathrm{EM}(\boldsymbol{X}^{\mathrm{m}}, K)$        ▷ See Section 2.5.6
12:
13:    # Obtain the $p$ GMM parameters through marginalizing
14:    for $p \in \{1, \dots, P\}$ do
15:        $\boldsymbol{\theta}^m_{p,k} = \{\boldsymbol{\mu}_{k,p}, \boldsymbol{\Sigma}_{k,p}, \pi_k\}_{k=1}^K \leftarrow \mathrm{Margin}(\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k\}_{k=1}^K, p)$
16:
17:    return $\boldsymbol{\theta}^m = \{\boldsymbol{\theta}_{p,k}\}_{1:N,1:p}$
```

4.3.3. MOTION SYNTHESIS

Motion synthesis is the process of generating the most likely data $\boldsymbol{x}^{\mathcal{O}}$ given the model parameters $\boldsymbol{\theta}^{\mathrm{m}}$, task-context $\boldsymbol{\theta}^{\mathrm{c}}$ and some input data $\boldsymbol{x}^{\mathcal{I}}$. Here, the input $\boldsymbol{x}^{\mathcal{I}}$ is not required to contextualize the TP-GMM. Yet, to synthesize movement, an external signal is required. In this chapter, the examples are limited to time-driven motions, i.e. they use time as input and pose as output. The use of alternative mappings such as required for autonomous dynamical systems [35, 120] are considered to be a valuable extension.

Algorithm 4.2 describes the procedure for motion synthesis. First, the TP-GMM that encodes the joint distribution of the state variables $\{\boldsymbol{x}^{\mathcal{O}}, \boldsymbol{x}^{\mathcal{I}}\}$ is contextualized using the parameters $\boldsymbol{\theta}^c$ (lines 2–5). This projects the P GMMs in the global coordinate system. Then, the conditional distributions $\mathcal{P}(\boldsymbol{x}^{\mathcal{O}}|\boldsymbol{x}^{\mathcal{I}})$ are computed for the P GMMs separately (lines 8–9). The contextualization and regression steps are changeable in order: one can regress the GMM in local coordinates and contextualize the result, or contextualize the GMM and regress its global projection. By projecting first, both input and output variables are contextualized. This could for example be used to adjust the motion execution by scaling the temporal signal.

The output of the conditioning gives P views of the desired state, namely $\hat{\boldsymbol{\mu}}_p^{\mathcal{O}|\mathcal{I}}$. The confidence of each output is given by the covariance $\hat{\boldsymbol{\Sigma}}_p^{\mathcal{O}|\mathcal{I}}$. The Gaussian product provides a way to fuse the different views while taking into account their confidence (line 12). The output of the Gaussian product is again a Gaussian. Its parameters give the desired state $\hat{\boldsymbol{\mu}}^{\mathcal{O}|\mathcal{I}}$ together with an expected covariance $\hat{\boldsymbol{\Sigma}}^{\mathcal{O}|\mathcal{I}}$.

Algorithm 4.2 Pseudo-code of the task-adaptive motion synthesis.

```
1:  function SYNTHESIS({$\boldsymbol{x}^{\mathcal{I}}$, $\boldsymbol{\theta}^c$, $\boldsymbol{\theta}^{\mathrm{m}}$ )
2:      # Create context-dependent model:
3:      for $p \in \{1,\cdots,P\}$ do
4:          $\hat{\boldsymbol{\mu}}_{k,p} = \Psi_{\boldsymbol{\theta}_p^c}\left(\boldsymbol{\mu}_{k,p}\right)$                ▷ See (4.5)
5:          $\hat{\boldsymbol{\Sigma}}_{k,p} = \Phi_{\boldsymbol{\theta}_p^c}\left(\boldsymbol{\Sigma}_{k,p}\right)$                ▷ See (4.6)
6:
7:      # Condition Individual Coordinate System:
8:      for $p \in \{1,\cdots,P\}$ do
9:          $\hat{\boldsymbol{\mu}}_p^{\mathcal{O}|\mathcal{I}}, \hat{\boldsymbol{\Sigma}}_p^{\mathcal{O}|\mathcal{I}} \leftarrow \mathrm{GMR}\left(\{\hat{\boldsymbol{\mu}}_{k,p}, \hat{\boldsymbol{\Sigma}}_{k,p}\}_{k=1}^K, \boldsymbol{x}^{\mathcal{I}}\right)$      ▷ See Section 2.5.7
10:
11:     # Merge contributions of each Coordinate System:
12:     $\mathcal{N}\left(\hat{\boldsymbol{\mu}}^{\mathcal{O}|\mathcal{I}}, \hat{\boldsymbol{\Sigma}}^{\mathcal{O}|\mathcal{I}}\right) = \prod_{p=1}^P \mathcal{N}\left(\hat{\boldsymbol{\mu}}_{\mathrm{p}}^{\mathcal{O}|\mathcal{I}}, \hat{\boldsymbol{\Sigma}}_{\mathrm{p}}^{\mathcal{O}|\mathcal{I}}\right)$      ▷ See Section 2.5.3
13:
14:     return $\hat{\boldsymbol{\mu}}^{\mathcal{O}|\mathcal{I}}, \hat{\boldsymbol{\Sigma}}^{\mathcal{O}|\mathcal{I}}$
```

4.3.4. Context Inference

Context inference estimates the context parameters $\boldsymbol{\theta}^{\mathrm{c}}$ given the context-independent model parameters $\boldsymbol{\theta}^{\mathrm{m}}$ and observed data $\boldsymbol{x}_n$. Unlike modeling and synthesis, the context inference problem has not yet been described in the light of TP-GMM.

We introduce the concept of context inference using an example given by Wilson and Bobick [121]. They exemplify parameterized gesture recognition using a fisherman who states: *"I caught a fish, and it was this big"*. The sentence is accompanied by a firm gesture indicating the size of the fish. The distance between the hand palms at the moment the fisherman says *"this"*, indicates the size of the fish—the parameter of the gesture. Note that this gesture can even be interpreted without speech. A competing fisherman following the conversation from a distance will understand the size of the fish solely by observing the gesture. The competitor is able to make this inference because he recognizes the gesture, knows the context, and—most importantly—understands which part of motions is key to infer the size of the fish.

This section presents a first attempt to parameter inference for TP-GMM. The approach is based on EM, which will be described first. Then, as context inference turns out to be complex, a strategy is proposed to avoid local optima.

EM for context inference

Context inference involves maximizing (4.1) with respect to the context parameters. As (4.1) contains hidden variables—the state activation—an EM-approach is proposed to solve the inference problem. As in any EM-algorithm, the E-step computes the responsibilities using the data log-likelihood, namely

$$\gamma_{n,k} = \frac{\pi_k \prod_p^P \mathcal{N}\left(\boldsymbol{x}_\mathrm{n} | \Psi_{\boldsymbol{\theta}_\mathrm{p}^\mathrm{c}}\left(\boldsymbol{\mu}_\mathrm{k,p}\right), \Phi_{\boldsymbol{\theta}_\mathrm{p}^\mathrm{c}}\left(\boldsymbol{\Sigma}_\mathrm{k,p}\right)\right)}{\sum_i^K \pi_i \prod_p^P \mathcal{N}\left(\boldsymbol{x}_\mathrm{n} | \Psi_{\boldsymbol{\theta}_\mathrm{p}^\mathrm{c}}\left(\boldsymbol{\mu}_\mathrm{i,p}\right), \Phi_{\boldsymbol{\theta}_\mathrm{p}^\mathrm{c}}\left(\boldsymbol{\Sigma}_\mathrm{i,p}\right)\right)}. \tag{4.11}$$

The computation of the E-step is straightforward, as it only involves function evaluations. Yet, for bad initial estimates of $\boldsymbol{\theta}^c$ the likelihood of the data $\boldsymbol{x}_n$ becomes very small. As a result, the denominator approaches zero, and could result in numerical instabilities. In Section 4.3.4, this will be addressed by adding a regularization term.

The maximization step optimizes the lower bound (A.3) with respect to $\boldsymbol{\theta}^\mathrm{c}$. This yields the optimization

$$\boldsymbol{\theta} = \arg\max_{\boldsymbol{\theta}^\mathrm{c} \in \mathcal{M}} \sum_{\boldsymbol{z}} \mathcal{P}\left(\boldsymbol{Z} | \boldsymbol{X}, \boldsymbol{\theta}^\mathrm{c,old}\right) \ln\left(\mathcal{P}\left(\boldsymbol{Z}, \boldsymbol{X} | \boldsymbol{\theta}^\mathrm{c}\right)\right), \tag{4.12}$$

which, given our model and observations $\boldsymbol{x}_n$, becomes

$$\boldsymbol{\theta} = \arg\max_{\boldsymbol{\theta}^\mathrm{c} \in \mathcal{M}} \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{n,k} \Bigg(\ln(\pi_k) + \underbrace{\sum_{p=1}^{P} \ln\left(\mathcal{N}\left(\boldsymbol{x}_\mathrm{n} | \Psi_{\boldsymbol{\theta}_\mathrm{p}^\mathrm{c}}\left(\boldsymbol{\mu}_\mathrm{k,p}\right), \Phi_{\boldsymbol{\theta}_\mathrm{p}^\mathrm{c}}\left(\boldsymbol{\Sigma}_\mathrm{k,p}\right)\right)\right)}_{\mathrm{E}(\boldsymbol{x}, \boldsymbol{\theta}^c, \boldsymbol{\theta}_k^m)} \Bigg). \tag{4.13}$$

Where $\mathrm{E}(\boldsymbol{x}, \boldsymbol{\theta}^c, \boldsymbol{\theta}_k^m))$ is introduced to simplify notation. It represents the context dependent part of the log-likelihood.

Generally, EM simplifies maximization of likelihood functions with latent variables [94]. For example, EM for parameter estimation of a Euclidean GMM has closed-form expectation and maximization steps. This simplification is attained enabled by splitting the optimization in two. This permits cancellation of the exponential function of the Gaussian.

Despite the cancellation of the exponential, the non-linear appearance of the context parameters $\boldsymbol{\theta}_p^c$ makes it unfeasible to solve the M-step effectively in closed form, or using a simple Gauss-Newton optimization as for Riemannian GMM. This is even true for a linear context parameterization and a Euclidean GMM. We demonstrate this by introducing the affine task parameters $\boldsymbol{\theta}^\mathrm{c} = \{\{\boldsymbol{A}_p, \boldsymbol{b}_p\}\}_{p=1}^P$. Using Table 2.1, the context transformations (4.5) and (4.6) simplify to

$$\Psi_{\boldsymbol{\theta}_p^c}(\boldsymbol{\mu}) = \boldsymbol{A}\boldsymbol{\mu} + \boldsymbol{b},$$
$$\Phi_{\boldsymbol{\theta}_p^c}(\boldsymbol{\Sigma}) = \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^\top.$$

Replacing these in $\mathrm{E}(\boldsymbol{x}_n, \boldsymbol{\theta}^c)$ we obtain,

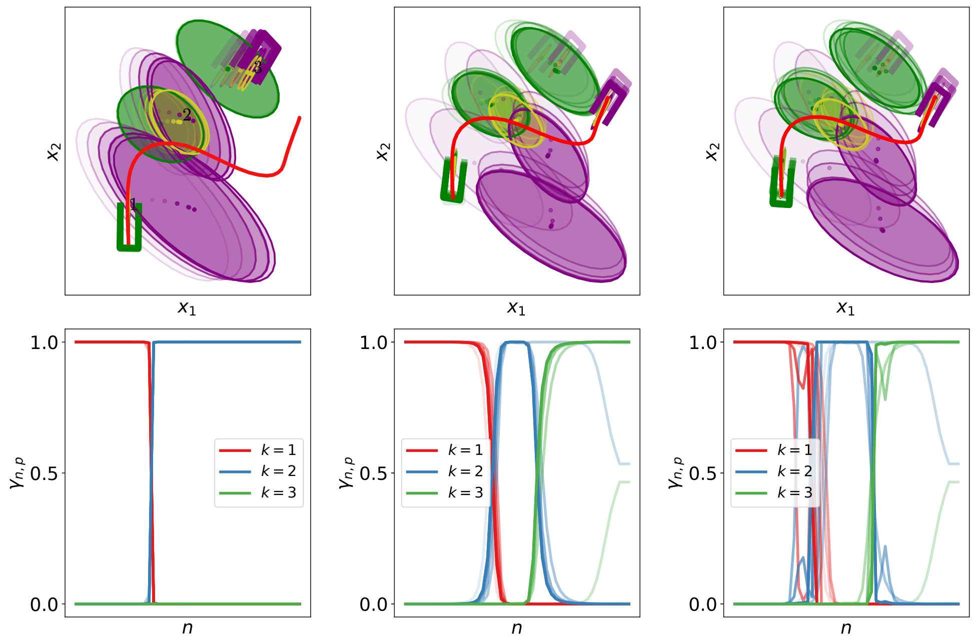
$$\mathrm{E}(\boldsymbol{x}, \boldsymbol{\theta}^c, \boldsymbol{\theta}_k^m) = c - \frac{1}{2}\sum_p^P \left((\boldsymbol{x} - \boldsymbol{A}_p\boldsymbol{\mu}_p + \boldsymbol{b}_p)\right)^\top \boldsymbol{A}_p \boldsymbol{\Sigma}^{-1} \boldsymbol{A}_p^\top \left(\boldsymbol{x} - (\boldsymbol{A}_p\boldsymbol{\mu}_p + \boldsymbol{b}_p)\right). \tag{4.14}$$

where $\boldsymbol{\theta}_k^m = \{\{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}_p\}_{p=1}^P$ are the model parameters of Gaussian k. The maximization of this expression is challenging because $\boldsymbol{A}_p$ appears until the fourth order and $\boldsymbol{A}_p$ is constrained to a manifold.

In order to perform the M-step, we resort to optimization on manifolds [67]. Manifold optimization is well suited for our approach, because the context manifold is known, as it follows from the context parameterization. When the context consists of rotations $\boldsymbol{A}$ and translations $\boldsymbol{b}$ in $3D$ space, our optimization manifold is $SO(3) \times \mathbb{R}^3$. When we consider more generic affine transformations, $\boldsymbol{A} \in GL(3)$ and $\boldsymbol{b} \in \mathbb{R}^3$, the optimization manifold is $GL(3) \times \mathbb{R}^3$.

E-step Regularization

EM is not guaranteed to converge to the global optimum. Instead, it is only guaranteed to convergence to a local optimum. This is not problematic when the optimization landscape contains few optima, and EM can be initialized close to a sufficient optimum. However, in the case of context inference for TP-GMM, various local optima can exist. We propose a regularization method that can improve EM for context inference.



(a) No Regularization (b) Regularization ($\lambda = 1e-2$) (c) Combined (it 1-3: $\lambda = 1e-2$, it 4-6 $\lambda = 0$)

Figure 4.2: Effect of regularization in context inference visualized on a peg-in-hole task. The top row visualizes the progress (increasing opacity) of the optimization in $2D$ space: the red line indicates the input data $\boldsymbol{x}_{1:N}$; the colored ellipsoids represent the Gaussians of the TP-GMM (colored according to their coordinate system). The bottom row visualizes the outcome of the E-step $\gamma_{n,k}$.

Each coordinate system of a TP-GMM contains a GMM. Typically, each GMM contains Gaussians with relatively small variance (invariant Gaussians), and Gaussians with relatively large variance (variant Gaussians). An invariant Gaussian indicates that context p is important for the task-representation, while variant Gaussians indicate the opposite. This behavior is evident in Figure 4.1, where the invariant directions govern the output of the product of Gaussians.

The E-step tends to assign invariant Gaussians low responsibility. This is reasonable since it is unlikely that an invariant Gaussian is responsible for $\boldsymbol{x}_{1:N}$ unless the data lies close to its center. Consequently, invariant Gaussians are likely to be ignored during the maximization step while they are most crucial in the estimation of the context parameters. Figure 4.2a exemplifies this scenario: the initial context estimate (the location and orientation of the purple and green holes) is such that almost no responsibility is assigned to the 3rd state. As a result, the estimated context parameters represent a bad local optimum.

To break this paradox we propose to regularize the E-step by replacing the model covariances $\boldsymbol{\Sigma}_{k,p}$ in (4.11) by

$$\hat{\boldsymbol{\Sigma}}_{p,k} = \boldsymbol{\Sigma}_{p,k} + \boldsymbol{I}\lambda, \tag{4.15}$$

with regularization factor $\lambda \in \mathbb{R}^+$. This 'inflates' the invariant Gaussians, ensuring the E-step assigns them responsibility. The effect of this regularization on the optimization outcome is visualized in Figure 4.2b. The bottom graph shows that the invariant Gaussians are now assigned responsibility. This indicates that the M-step takes them into account. As a result, the EM outcome shows a more reasonable estimate of the context parameters (top graph).

Once the regularized optimization converged, the regularization factor can be set to zero. This allows further improvement of the estimated context. The results of this combined approach are shown in Figure 4.2c.

To objectively compare the results of the two optimization options, we review their cost (the lower bound (A.3)) which are visualized in Figure 4.3. The results confirm that the regularization assisted in reaching a better optimum. But the results also demonstrate that the regularized approached prevents reaching the true local minimum. This is caused by the relatively high penalty that the regularized solution obtains for small misalignments of the invariant Gaussians. The combined solution takes best of both world by further refining the solution of the regularized solution. On the downside, we need to define a heuristic that allows the algorithm to determine when to switch from regularized to non-regularized optimization.

4.4. EVALUATION

4.4.1. WRITTEN LETTERS, A SINGLE FRAME EXAMPLE

The actions of modeling, synthesis and inference are demonstrated on a 2D data set of hand written characters as visualized in Figure 4.4. This data serves well for illustrative purpose, because they are easily visualized and contain a variety of motions. Both affine and rigid-body transformations are considered as context parameterization for this data set. The context describes the pose or shape of the letter with respect to the observer.

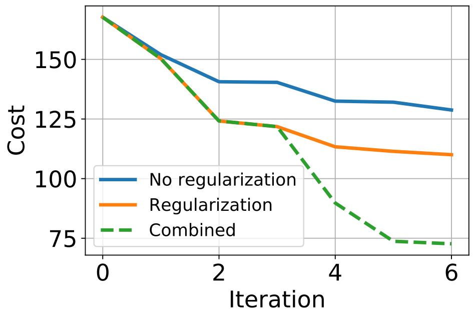


Figure 4.3: Cost comparison

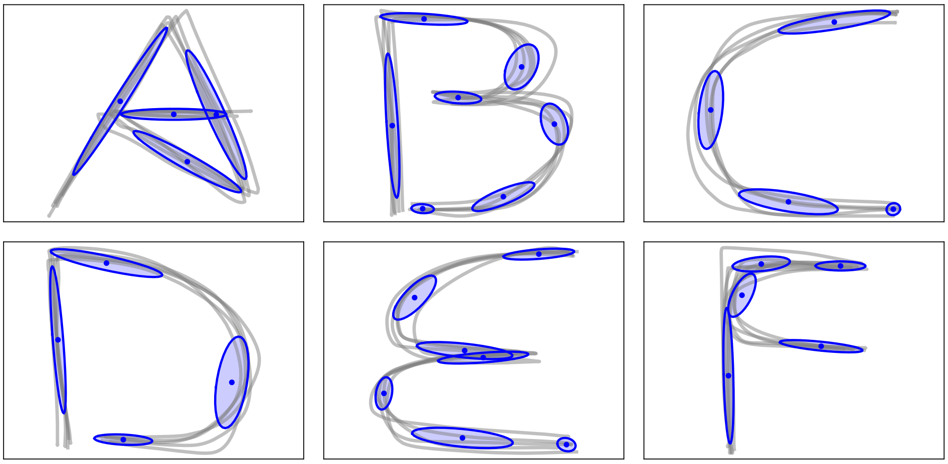


Figure 4.4: Demonstrations data and estimated GMMs of the letter data set.

MODELING

The data set consists of 6 letters, each demonstrated 5 times. Each demonstration is resampled to 200 data points $\{t, \boldsymbol{x}\} \in \mathbb{R} \times \mathbb{R}^2$, and its temporal signal is normalized such that it lies in the range $[0, 1]$. During the demonstration phase the context is held fixed, i.e. per letter each demonstration is performed in a coordinate system of equal shape and pose. As a result, the variability observed among demonstrations relates to shape and pose variations that are allowed within a specific context. From a modeling perspective it makes no difference whether these demonstrations are given in single or multiple contexts. When there is only one coordinate system (i.e. $P = 1$), the demonstration data are transformed and modeled in a single context independent frame. For example, if the letter 'A' would have been demonstrated in contexts of different position and rotation, their projections into the context independent space will overlap as if they were given in a single position and rotation. In contrast, with multiple coordinate systems ($P > 1$), it is important to demonstrate the task in a variety of contexts. By changing the context, the importance of the individual coordinate systems can be discovered. This influences the ability to generalize to new situations, as will be discussed in Section 4.4.2.

The demonstration data of each letter are used to estimate a joint probability distribution $\mathcal{P}(t, \boldsymbol{x})$ that is represented using a GMM. The number of Gaussians in each GMM was manually set (4 (A), 7 (B), 4 (C), 4 (D), 7 (E) and 5 (F)). The parameters were estimated using EM, and initialized using K-bins (see Appendix A.1). The resulting models and the demonstration data are visualized in Figure 4.4.

SYNTHESIS

After modeling, the letters can be synthesized in a variety of contexts. Figure 4.5 demonstrates synthesis examples for rigid-body and affine transformations. The letters are synthesized through contextualizing and conditioning the modeled distribution. The particular contexts are given by

$$\boldsymbol{A}_{\mathrm{SE}(2)} = \boldsymbol{R}_{\mathrm{x}}(\theta) \qquad , \ \boldsymbol{b}_{\mathrm{SE}(2)} = \boldsymbol{v}, \tag{4.16}$$

$$\boldsymbol{A}_{\mathrm{Aff}(2)} = \boldsymbol{R}_{\mathrm{x}}(\theta)\mathrm{diag}(\boldsymbol{s}), \ \boldsymbol{b}_{\mathrm{Aff}(2)} = \boldsymbol{v}, \tag{4.17}$$

where $\boldsymbol{R}_{\mathrm{x}}(\theta) \in \mathrm{SO}(2)$ is the 2D rotation parameterized by θ

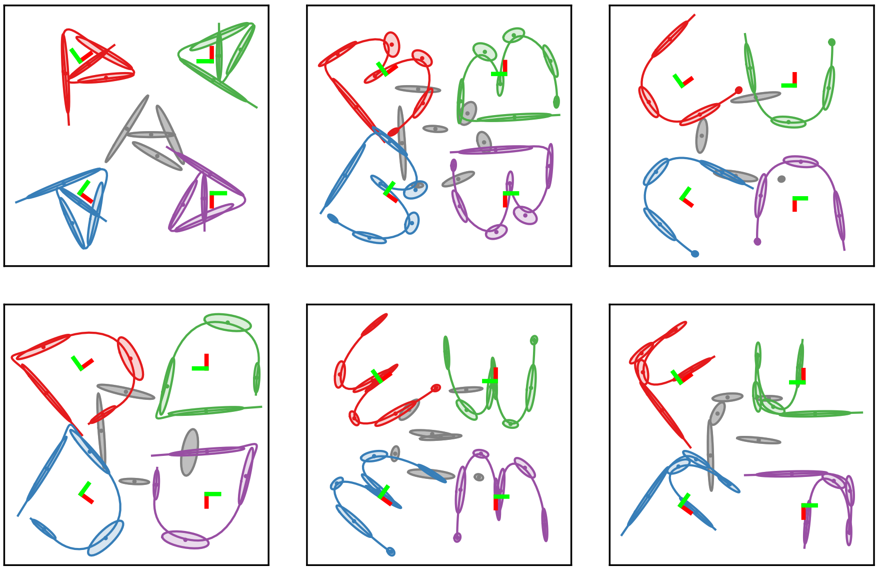
$$\boldsymbol{R}_{\mathrm{x}}(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}, \tag{4.18}$$

$\boldsymbol{v} \in \mathbb{R} \times \mathbb{R}^2$ a translation, and $\boldsymbol{s} \in \mathbb{R}^3$ a scaling. Table 4.4 presents the parameter values used for the synthesis examples in Figure 4.5. Note that the rotation and scaling are 3D values because they transform joint distribution of time and position. This formulation also allows the modification of execution time by changing the temporal scaling (first entry of $\boldsymbol{s}$).

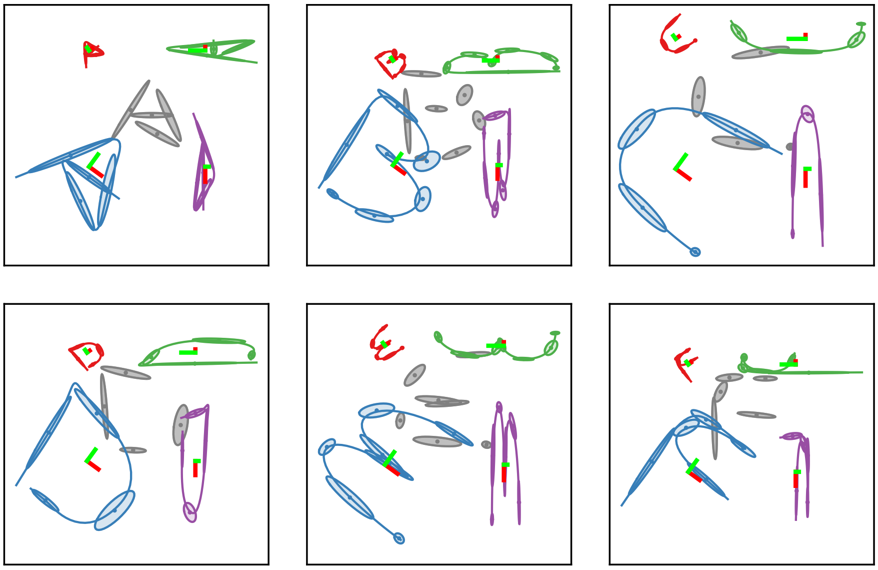
	θ	$\boldsymbol{v}$	$\boldsymbol{s}$
H_1	$\frac{\pi}{5}$	$(0, [-10, 10]^\top)$	$[1, 0.3, 0.3]$
H_2	$-\frac{\pi}{5}$	$(0, [-10, -10]^\top)$	$[1, 1.3, 1.3]$
H_3	$\frac{\pi}{2}$	$(0, [10, 10]^\top)$	$[1, 0.3, 1.3]$
H_4	$-\frac{\pi}{2}$	$(0, [10, -10]^\top)$	$[1, 1.3, 0.3]$

Table 4.4: Parameters that describe the rigid-body and affine transformations used in synthesis of the letter reproduction examples.

In practice, we see that the GMM can be properly transformed under both rigid-body and affine transformations. The transformations preserve the spatial and temporal relations that are encoded in the original GMM. The latter is confirmed by the fact that the expected values $\mathbb{E}(\mathcal{P}(x|t))$, computed using GMR, clearly reproduce the encoded letters.



(a) Rigid-body transformations



(b) Affine transformations

Figure 4.5: Synthesis of 6 written letters using two different context parameterizations. In (a), 4 rigid body transformations are applied to the letter data set. (b) Applies 4 affine transformations to the rigid body data set. Each graph displays the original model in gray and the 4 transformed models in color. Additionally, the regressed motion $\mathcal{P}(x|t)$ is displayed using the thick lines of corresponding color.

INFERENCE

Context inference can be used to estimate the context under which a written letter is drawn. Here, the inference capabilities of the approach presented in Section 4.3.4 is assessed. EM for context inference requires the motion data for which we want to estimate the context, the context-independent model, an initial guess of the context parameters, and the tuning parameters of the algorithm. The input data are given by the synthesis results displayed in Figure 4.5. The true context thus corresponds to (4.16) with the parameters specified in Table 4.4. As initial guess we take the identity transformation: $\boldsymbol{A} = \boldsymbol{I}_3$ and $\boldsymbol{b} = (1, \mathbf{0})$. EM is regularized ($\lambda = 10$) for the first 3 steps, and continued unregularized ($\lambda = 0$) until convergence.

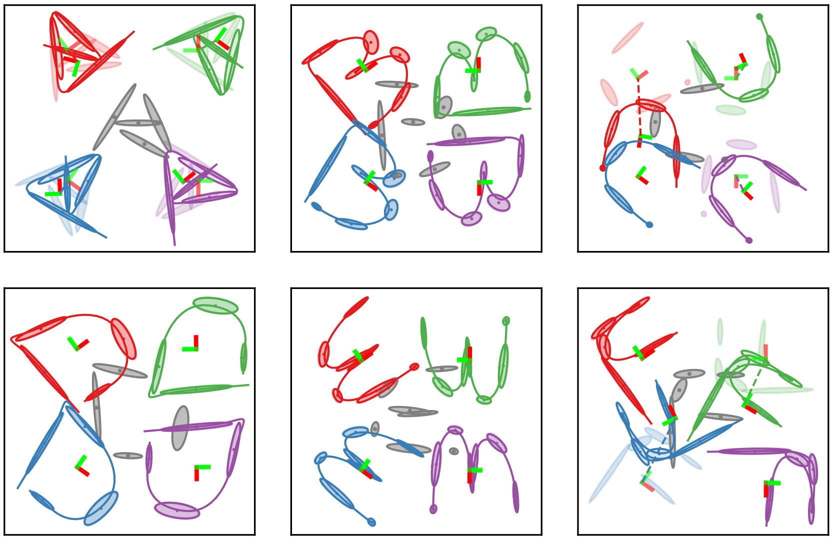
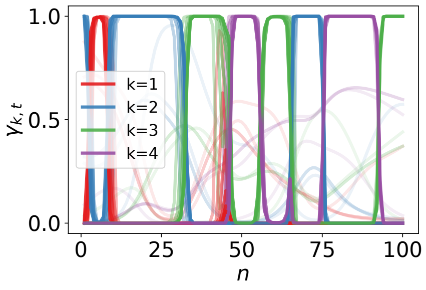


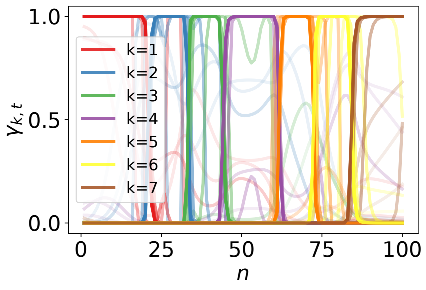
Figure 4.6: Context inference for 2D written letters. Each of the sub-figures displays the context inference results for 4 different contexts (colored red, green, blue and purple). The initial context estimate is visualized using the gray contextualized GMM in the center of each sub-figure. The spatial misalignment between the true contexts (transparent) and the estimated contexts (opaque) are indicated by the dotted lines of corresponding color. The rotational misalignment is indicated by the red and green orthogonal lines.

The outcome of the context inference for all letters is visualized in Figure 4.6. The inference was successful for the letters B, D, and E. Partially successful for the letters C and F, and not successful for the letter A. In the case of the letter A, the inference was able to approximate the required translation, yet unable to identify the required rotation. Figure 4.7 demonstrates the cause of this failure. It shows the state responsibility assignments for a successful and unsuccessful inference attempt. In the successful case, the states are 'activated' in sequence, while in the other case the states are not activated in appropriate order.

Using the context-inference, can also be used for classification. The goal of classification is to identify to which class a certain movement belongs. Given a query movement, classification requires two steps. First, context inference is performed on each



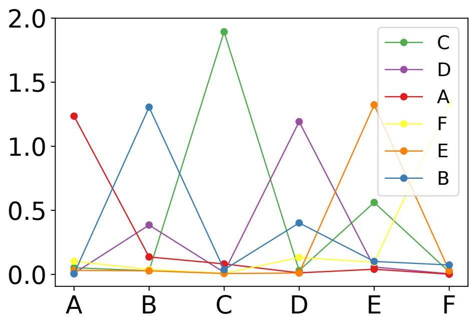
(a) Responsability assignment for the letter A.



(b) Responsability assignment for the letter B.

Figure 4.7: State responsibilities during context-inference for unsuccessful inference (a) and successful inference (b). The increasing opacity indicate the progress of the EM.

class model (letter model in this case). Then, the likelihood of the query data is evaluated for each context-dependent model. The model with the highest likelihood indicates the class of the query data. Figure 4.8 gives the classification results for the letter data set. For this particular transformation, the classification results are successful for all letters (see Figure 4.9). However, the classification relies on a properly estimated context. In case inference is unsuccessful for the true class model, classification is likely to fail. Such situations could be detected by requiring a minimum likelihood for classification. If none of the classes reaches a likelihood beyond this minimum, the system could indicate classification is not reliable.



(a) Likelihood per letter

Figure 4.8: Classification results. The colored dots indicate the data likelihood after context-inference per letter.

4.4.2. PEG-IN-HOLE, A TWO FRAME EXAMPLE

The context-adaptive representation enables generalization of demonstrated tasks to new situations. This task was already used as an illustrative example, but in this section it is evaluated in more detail. The goal of the task is to move from one hole into another.

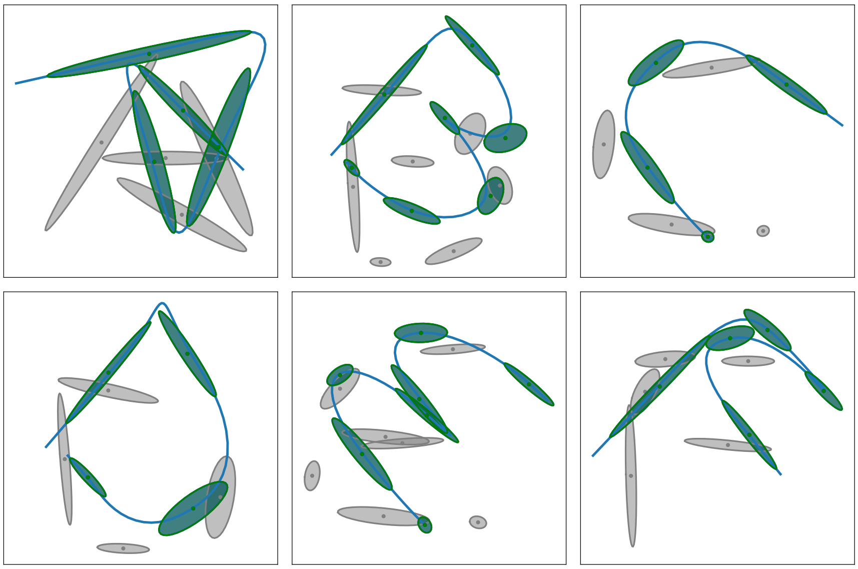


Figure 4.9: Context inference results obtained during classification. The blue lines indicate the input data of the inference algorithm. The figures show the letter models using the initial context prior to inference (gray), the models contextualized using the true context parameters (green), and the models contextualized using the inferred context parameters (blue). The blue and green colored models appear merged, as the inferred context lies very close to the true context.

Here, the departure and arrival at the start and goal holes have constrained directions. As the position and orientation of both holes can vary, it is desirable to generalize the demonstrated task to unseen context.

MODELING

The task context is parameterized by the rigid-body transformations of start and goal holes. During the demonstration phase, four examples are given of the task while recording the temporal $2D$ spatial information. Each demonstration, the position and orientation of the goal are changed. This allows the algorithm to discover the importance of the different frames.

The demonstration data are projected and modeled in the context-independent coordinate systems using the algorithm described in Section 4.1. This GMM is defined on the manifold $\mathbb{R}\times\mathbb{R}^2\times\mathbb{R}^2$, corresponding to the temporal information and spatial context-independent data. The demonstration data and the model are visualized in Figure 4.10.

This model captures the essence of context-independent encoding: invariant features. The demonstration data shows different variance when considered from different perspectives. At the start of the motion low variance is observed in the start frame, and large variance in the goal frame. While at the end of the motion the opposite is observed.

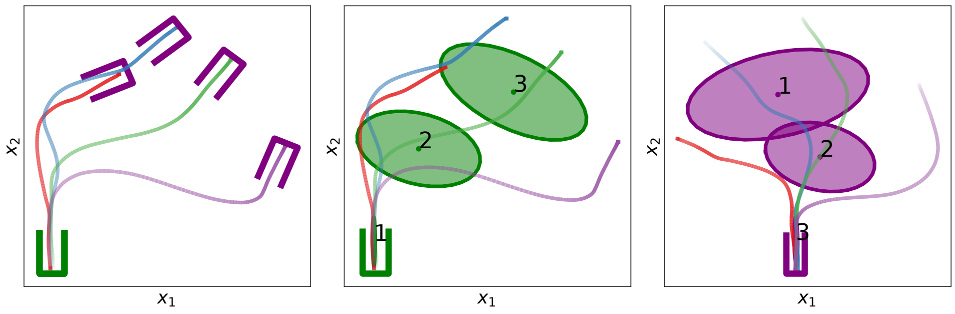


Figure 4.10: Modeling of a 2D peg in hole task. Left: context-dependent demonstrations of the peg-in-hole task. Middle&Right: The demonstration data projected and modeled in the start and goal coordinate systems. Each demonstration is uniquely colored, and the line opacity indicates its temporal progress.

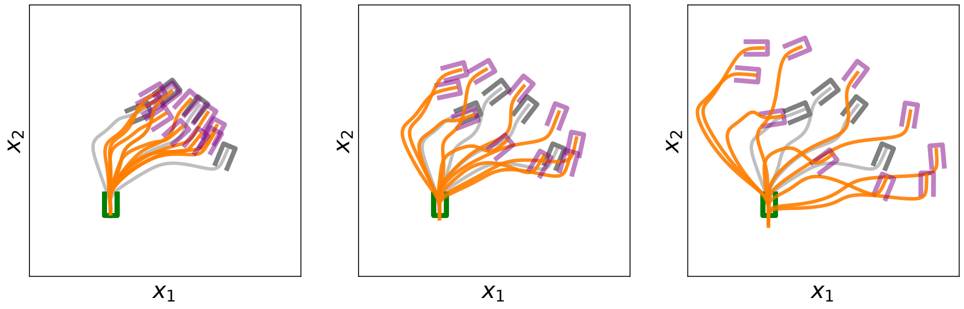
SYNTHESIS

TP-GMM allows the synthesize of motion in unseen context. This is achieved using the steps described in Section 4.3: contextualizing the model, performing GMR in the individual frames, and finally combining the GMR results using the Gaussian product. This section demonstrates the generalization capabilities and limitations of motion synthesis.

The ability of TP-GMM to generalize to unseen context is limited by the variability that was observed during the demonstrations. To assess this ability we define a measure of generalization. The proposed measure quantifies generalization difficulty by the likelihood that a context has been observed before. This requires a context distribution, which is approximated using a single Riemannian Gaussian on the context manifold $\mathrm{SO}(2)\times\mathbb{R}\times\mathrm{SO}(2)\times\mathbb{R}$, i.e. $\boldsymbol{\theta}^c \sim \mathcal{N}(\boldsymbol{\Sigma}^c, \boldsymbol{\mu}^c)$. The parameters of the context distribution are estimated from the contexts encountered during the demonstration phase.

With the Gaussian-based measure, generalization difficulty depends on the likelihood of a context to belong to this distribution. Figure 4.11 illustrates peg-in-hole synthesis for context instances that lie 1, 2, and 3 standard deviations from the mean context. Each sub-figure thus illustrates examples of equal difficulty. These results show that TP-GMM is able to generalize the task to likely context ($\sigma \leq 1$). Furthermore, as the generalization difficulty rises, conflicts arise between the synthesized data and the task constraints. This is most apparent in Figure 4.11c, where the synthesized data crosses the borders of the start and goal hole.

The constraints are violated because the Gaussian product puts too much emphasis on the information encoded in the goal frame at the start of the motion, and vice versa for the end of the motion. The trade-off made by the Gaussian product relies on relative variability between the start and goal frame. When a context instance lies far from the context observed during demonstrations, the relative variability of model is too scarce to fully meet the task constraints. Yet, the resulting motion reflects the demonstrated task constraints, a feature which could not be achieved with standard regression techniques such as GMM-GMR or DMP. Furthermore, the synthesis reliability can be assessed using the presented difficulty measure. In active learning scenarios this measure can be used to trigger a demonstration requests when a robot encounters situations that cannot be performed reliably [122, 123].



(a) $\sigma = 1$ (b) $\sigma = 2$ (c) $\sigma = 3$

Figure 4.11: Synthesis of the peg-in-hole task for samples with a standard deviation σ from the mean context $\boldsymbol{\mu}^c$. The samples are equally spaced over the contour of equal probability spanned by the largest two eigen components of the context covariance $\boldsymbol{\Sigma}^c$. The synthesized data is visualized in yellow. The start and goal positions of the samples are visualized in green and purple, respectively. The gray lines and holes visualize the original demonstrations and their context.

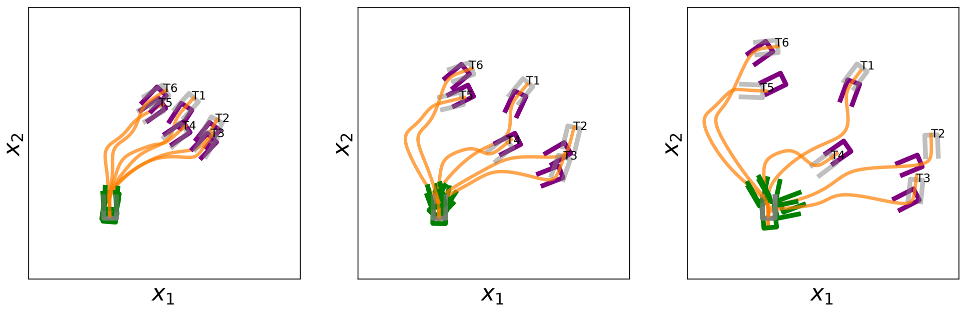
INFERENCE

Finally, context inference is performed on the peg-in-hole task. The objective is to estimate the context parameters $\boldsymbol{\theta}^c$, the position and orientation of the start and goal, given an input trajectory $\boldsymbol{X} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\}$. Inference is performed with regularization as described in Section 4.3.4. The E-step is regularized during the first three EM steps ($\lambda = 0.1$), after which the regularization is removed (i.e. $\lambda = 0$) and EM is repeated until convergence. EM is initialized at the mean context $\boldsymbol{\mu}^c$.

The TP-GMM of the peg-in-hole task encodes both temporal and spatial information. In this example, inference only relies on the spatial information. This is more practical as it does not require temporal rescaling of the input data. Yet, this makes the inference more challenging, as the individual frames of the TP-GMM can no longer be temporally aligned.

Similarly to synthesis, the inference quality is assessed using context parameters of varying difficulty. Based on the context distribution, 3×6 context instances are selected. 6 instances equally spaced over contours of equal probability at 1, 2 and 3 standard deviations from the mean context. These context instances form the ground truth. Input data $\boldsymbol{X}$, required to validate the inference approach, is generated by synthesizing the most-likely motion for each context instance as described in Section 4.4.2.

Figure 4.12 demonstrates the inference results, i.e. the estimated context $\boldsymbol{\theta}^c$ given the TP-GMM, and the input data $\boldsymbol{X}$. Figure 4.13 visualizes the error between the ground truth and the estimated context. Similarly to the synthesis example, for likely context instances the results are satisfying. However, the results degrade as the likelihood decreases. When a context distribution is available, the reliability of the inference result can be assessed by its likelihood.



(a) $\sigma = 1$ (b) $\sigma = 2$ (c) $\sigma = 3$

Figure 4.12: Inference of the peg-in-hole task for samples with varying standard deviations from the mean context $\boldsymbol{\mu}^c$. The samples are equally spaced over the contour of equal probability spanned by the largest two eigen components of the context covariance $\boldsymbol{\Sigma}^c$. The inferred context is visualized in purple and green. Gray holes visualize the context used to synthesize the orange input data.

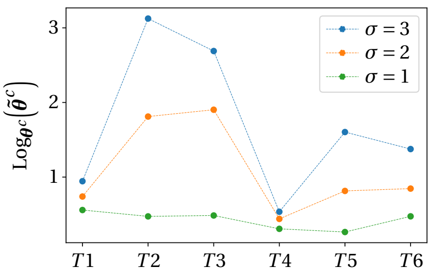


Figure 4.13: Visualization of the context inference errors observed in Figure 4.12 for different levels of difficulty ($\sigma = 1, 2$ and 3).

4.5. Applications

This section presents applications of the context-adaptive approach. The applications include two robotic experiments, and a more artificial example which clearly demonstrates the capabilities of affine contexts. All applications in this section consider 3D position and orientation and leverage the Riemannian framework in a context-adaptive setting.

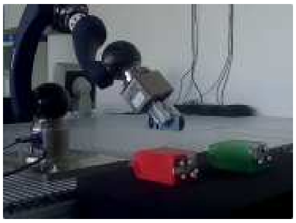
Table 4.5 gives an overview of the task manifold, context manifold, modeling and synthesis methods used for the different experiments. The Pick & Place and Car Door tasks are the outcome of a collaboration within the SMART-E project [57][1].

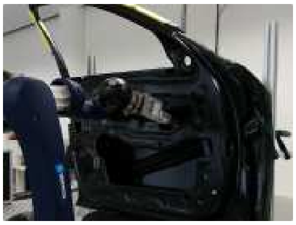
This section gives an overview of the experimental results, and presents insights gained from these experiments. Detailed descriptions of the experiments are given in Appendix B.

Task	**Data**	**Context**	**Modeling**	**Synthesis**	**Appx**
Pick & Place	$\mathbb{R}^3 \times \mathcal{S}^3 \times \mathbb{R}$	$SE(3) \times SE(3)$	time-based + EM	GMR + Product	B.1
Car door	$\mathbb{R}^3 \times \mathcal{S}^3$	$SE(3)$	time-based + EM	GMR	B.1
Box Tape	$\mathbb{R}^3 \times \mathcal{S}^3$	$Aff(3) \times SE(3)$	time-based + EM	GMR + Product	B.2

Table 4.5: An overview of the context-adaptive experiments.

4.5.1. Task Descriptions


(a) Pick & Place


(b) Door Task

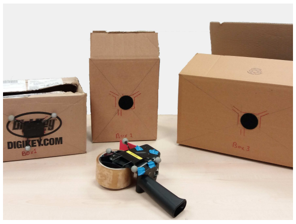

(c) Box Tape

Figure 4.14: Overview of the tasks that were programmed using the context-adaptive approach presented in this chapter.

Figure 4.14 illustrates the 3 context-adaptive tasks that are discussed in this section. The Pick & Place (Figure 4.14a) and Door Task (Figure 4.14b) were used to validate a modular approach to flexible automation (see Appendix B.1 for details). Both tasks are recorded using a Vicon motion tracking system, and reproduced using a Schunk manipulator.

The objective of the Door Task is to track a profile inside the door with sufficient pressure and correct orientation. In practice, such a motion is used to attach fabric onto

[1]The collaborations in the SMART-E project are with Andrea Giusti, Esra Icer and Aaron Pereira of the Technical University of Munich (TUM).

the door. Unlike the other applications discussed in this section, it only considers a single local coordinate system: the rigid-body pose of the door. The PICK & PLACE requires the move of an object from the green to the red box. In this tasks, the rigid-body pose of the boxes comprise the task context. The PICK & PLACE task involves a grasp and release action. This binary signal is represented on $\mathbb{R}$. Together with the end-effector pose of the robot $(\mathbb{R}^3 \times \mathcal{S}^3)$, this yields the demonstration manifold $\mathbb{R}^3 \times \mathcal{S}^3 \times \mathbb{R}$.

The BOX TAPE task (Figure 4.14c) involves closing a box using a tape tool. This requires a taping motion that goes across the closing seam on top of the box, and is started and stopped about half-way the adjacent sides. The task context comprises the pickup location of the tape tool, and the location and shape of the box. These are parameterized by a rigid-body and affine transformation, respectively. Using the affine context parameterization, the skill can be generalized to boxes with different sizes. The data for this experiment are recorded using a marker-based motion capture system. Synthesis of the motion is performed in simulation.

4.5.2. DEMONSTRATION DATA & MODELING

Task	**#Dem**	$\mathcal{M}$	K
AXIS BRUSH	4	$\mathbb{R} \times (\mathbb{R}^3 \times \mathcal{S}^3) \times (\mathbb{R}^3 \times \mathcal{S}^3)$	6
BUTTON PUSH	5	$\mathbb{R} \times (\mathbb{R}^3 \times \mathcal{S}^3) \times (\mathbb{R}^3 \times \mathcal{S}^3)$	6
PICK & PLACE	9	$\mathbb{R} \times (\mathbb{R}^3 \times \mathcal{S}^3) \times (\mathbb{R}^3 \times \mathcal{S}^3) \times \mathbb{R}$	6
CAR DOOR	3	$\mathbb{R} \times (\mathbb{R}^3 \times \mathcal{S}^3) \times (\mathbb{R}^3 \times \mathcal{S}^3)$	15
BOX TAPE	6	$\mathbb{R} \times (\mathbb{R}^3 \times \mathcal{S}^3) \times (\mathbb{R}^3 \times \mathcal{S}^3)$	9

Table 4.6: Overview of empirically chosen parameters for demonstrating and modeling.

To program a task, it was demonstrated several times. It is not straight forward to determine the exact amount of demonstrations to guarantee proper imitation. For the context-adaptive approach the set of demonstrations should allow the discovery of variant and invariant features in the different coordinate systems. When the data set is too small, these features will not be properly discovered. Requiring a large amount of demonstrations might annoy the user. In our experiments, each demonstration is given in a different context. The number of demonstrations for each task, given in Table 4.6, followed empirically from the context variation we expected in each task. The context selection was based on basic knowledge of the task, and the understanding that TP-GMM needs to discover invariant features. Although the context variation was intentional, it was not pre-engineered (we did not pre-analyze contexts to attain maximum variance/invariance). As the CAR DOOR task, only involves a single coordinate system, no variant or invariant features are required for generalization. In this case a small number of demonstrations was used to omit potential inaccuracies produced during the demonstrations.

Each demonstration is recorded separately. After demonstration excess data (recorded before and after each demonstration) was trimmed by the demonstrator using a graphical user interface. In addition, the temporal signals were linearly rescaled to make them range from 0 to 1. These simple pre-processing steps improve temporal alignment of the demonstration data, and therefore contribute to a better model. No additional filtering or signal processing steps were involved.

After demonstration, the data are modeled in Riemannian GMM. Here, we rely on the algorithm described in Section 4.3.2, which projects the demonstration data in the local coordinate systems and fits a GMM of K states using EM. The number of states of each GMM was empirically chosen (see Table 4.6). The EM algorithm converged for all tasks within 100 iterations.

4.5.3. SYNTHESIS

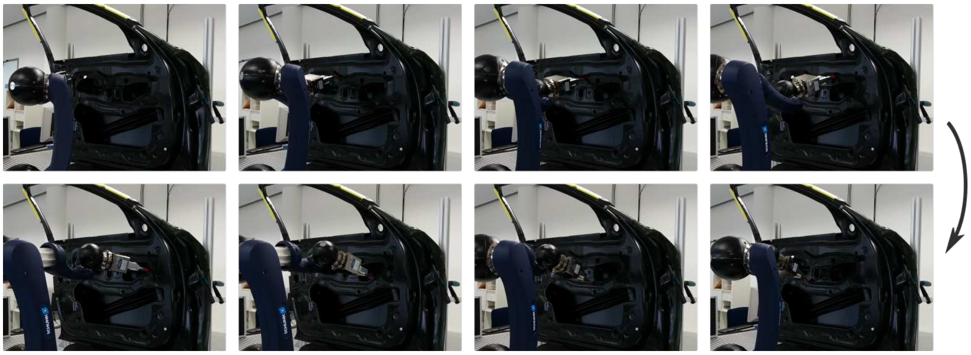


Figure 4.15: A typical reproduction of the door task

The different tasks are synthesized following Algorithm 4.2. The CAR DOOR model contextualization is based on the new door pose. Using the contextualized model, the task motion is synthesized. This is achieved using GMR with time as input, and position and orientation as output. To keep the joint velocities within the hardware limitations, the generated trajectory is dynamically rescaled in time. This process can be automated, as discussed in for example [124]. The reproduction is displayed using snapshots in Figure B.6, and in the video. They show a smooth trajectory which complies with the demonstrations. Both position and orientation aspects of the demonstrations are preserved and properly adapted to the new context (door pose).

The PICK & PLACE is reproduced by contextualizing its model to the new pick and place poses. Similar to the CAR DOOR, the synthesized trajectories are dynamically rescaled to meet joint velocity limits. Snapshots of the PICK & PLACE task are given in Figure 4.16, and the full reproductions are featured in the video. The reproductions demonstrate that the robot can adapt its motion to the new contexts. However, during placement the robot slightly pushes the object onto the place location, a behavior which not has been intentionally demonstrated. Yet, the model contains unintended correlation among the rotational and spatial dimensions. Practically, such behavior can be avoided using shrinkage regularization, as described in Appendix B.1.3. Effectively, this technique lessens the correlation.

The BOX TAPE task is synthesized in seen and unseen contexts. The results are visualized in Figure 4.17. The figure shows how the GMM adapts to the shape of the box. Note the relative horizontal and vertical stretch of the GMMs in Figures 4.17a and 4.17b. They

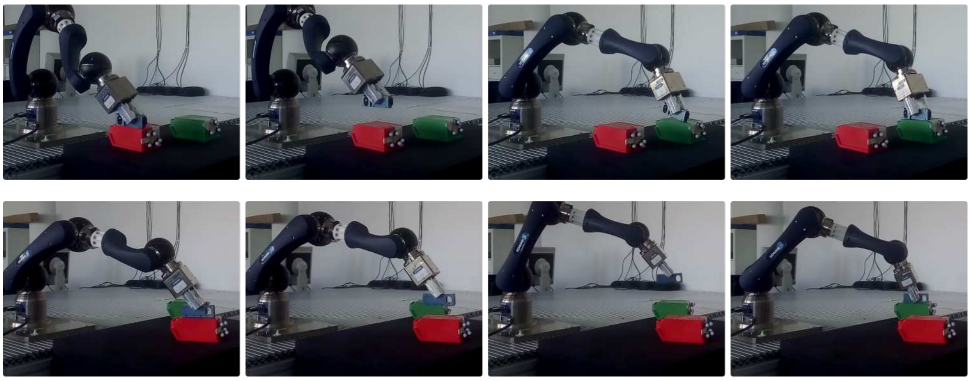
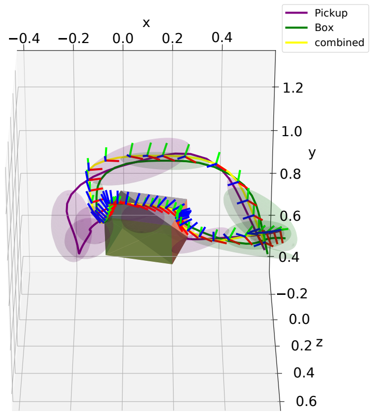
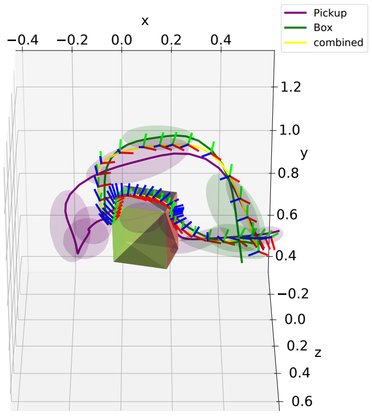


Figure 4.16: Snapshot of 2 reproductions of the Pick & Place task.

show that GMM adapts to the box shape. Furthermore, note that the encoded orientation of the end-effector changes in a similar fashion for both box shapes.



(a) Reproduction on box 1



(b) Reproduction on box2

Figure 4.17: Box taping reproductions in a previously encountered context (a), and new context (b). The 3D figures(a,b) show the position outcome of GMR for the box and pickup frame in green and purple, respectively. The outcome of the product of Gaussians (yellow) also demonstrates the resulting orientation using the 3 colored orthogonal lines. The cubic shape indicates the pose of the box.

4.6. Discussion

In this chapter, we presented a context-adaptive framework for modeling, synthesis and inference of task-space motion. The framework completes the previously proposed TP-GMM by presenting a method for context inference, and extends it from task-space position to task-space pose. Additionally, we presented an overview of context types that are compatible with the task-parameterized approach.

This chapter highlights the benefit of affine context in modeling and synthesis of motion. Compared to rigid-body transformations, affine transformations admit object scaling in addition to object position and rotation. This enables well-supplied generalization capabilities. These have been demonstrated by the experiments presented in Section 4.4.1 and Section 4.5. These experiments showed that a (Riemannian) GMM can be contextualized using affine transformations; that it admits the use of both position and orientation in 2D and 3D; and, that it can combine different types of context-parameterizations (in our case affine and rigid-body transformations).

TP-GMM extracts context importance from demonstration data; this importance is given by variant and invariant patterns in the different coordinate systems. TP-GMM only requires the definition of candidate objects, their importance is extracted from demonstration. Furthermore, TP-GMM is indifferent about the way coordinate systems are defined on objects. It only requires this definition to be consistent (once a coordinate system is set for an object, its definition should not change). This is an advantage over other context-adaptive approaches [22, 27, 28, 31, 107]. These require manually defined task features, such as goal position [27, 28, 107], via-points [22] or object height [31]. The ability to discover important features from demonstration data makes TP-GMM more generic. For example for a peg-in-hole task, our approach only requires a coordinate system associated with the object that contains the hole, while approaches like DMP and ProMP require specific coordinates of the hole. We have to note that this generality comes at a cost. The task-features are not discrete, instead they emerge from a trade-off performed by the product of Gaussians. This limits the generalization capabilities, as was demonstrated in the peg-in-hole task. There, the extrapolation attempts resulted in collision between the peg and the hole. In future work, this could be resolved through discretisation of invariant patterns in such a way that they become hard constraints (e.g. through modification of the covariance matrices by setting small eigenvalues to zero, or large eigenvalues to infinity).

In our evaluations we empirically set the number of Gaussians K of the TP-GMM. Although this thesis does not focus on automatic model selection, we acknowledge this is required for a fully functioning PbD system. We attempted selecting the number of Gaussians using the Bayesian Information Criterion (BIC) [125], but found this method to easily over-estimate the number of Gaussians. This is potentially explained by the fact that demonstration data do not contain well separated clusters. Further research is required to find alternative measures to determine the optimal number of clusters. Possible directions could lie in the use of Dirichlet processes as used in [126, 127] .

In our experiments, we achieved object recognition using marker-based motion tracking systems. Such system might not be available in many scenarios. In this light, point-cloud matching is considered a viable technique that can be linked to the context parameterization. Such application would share similarities with the work of Brandi *et al.* [109].

This chapter discussed context inference for TP-GMM for the first time. Context-inference enables the robot to recognize in what context a motion is executed. This has potential application in intention recognition (classification of action) and non-verbal communication (e.g. communicating object sizes through hand gestures). Unlike modeling and synthesis, context-inference involves a complex optimization. We presented and evaluated an EM-based approach to solve it. One of the main challenges faced by the EM algorithm is responsibility assignment—i.e. the process that determines which data points are most relevant to infer context. Although we proposed a regularization method which lessens this problem, our experimental evaluation showed that the proposed approach did not always infer context properly. We expect that inference can be improved by explicitly taking into account the state activation order using a Hidden Markov Model (HMM) or Hidden Semi-Markov Model (HSMM) [128], as this provides the algorithm additional temporal information. With an improved EM procedure, the presented approach is readily extendable to richer context (i.e. affine transformations), and 3D data which included both position and orientation. Such an extension solely requires the modification of the EM objective function.

5
Programming Shared Control by Demonstration

In chapters 2–4, we presented a Riemannian approach for Programming by Demonstration (PbD) which has geometry-aware, synergetic and context-adaptive properties. In this chapter, we use these properties to generate shared control strategies for teleoperation.

5.1. Introduction

Teleoperation has been a key drive for robotics research. It stems from the pragmatic need to perform tasks in remote environments. These tasks occur in a broad application domain, ranging from deep sea to outer space. Traditionally, the robot motion is directly controlled by the operator. In such systems, the performance of the operator is improved by increasing the *transparency* of the teleoperation system, and by providing her with a *feeling of presence* [129].

Virtual assistance can further improve the teleoperator performance [130, 131]. Such systems share or delegate task execution to an assistive system, with the aim of reducing the operator's cognitive load. The reduced cognitive load could allow the teleoperator to perform teleoperation for longer periods of time.

Often, shared control approaches make use of *virtual fixtures*. These constraint the manipulability of a robot by restricting the system state to a defined space (area/volume). Metaphorically, a virtual fixture can be seen as a ruler which aids users to draw straight lines [132, 133]; its use significantly improves task performance [132, 134], and reduces mental workload [133].

In teleoperation, virtual fixtures can prevent the remote robot, the slave, to collide with the environment. For another example, virtual fixtures can constrain the orientation of a tool to remain perpendicular to a wall. This way, the operator controls the position of the end-effector while the assistive system handles its orientation. Ideally,

The contents of this chapter have been submitted to IEEE Robotics and Automation Letters (RA-L).

the operator performs the intelligent part of a task—deciding where to drill—while the assistance handles the trivial part—the perpendicular constraint.

Virtual fixtures are often hand-coded as attractors or repulsors that drive the system towards or away from a certain state [133]. However, the wide variety of tasks that can benefit from virtual fixtures make manual coding a daunting task. We argue that the user-friendly interface offered by PbD [4] can alleviate this problem, and propose to program virtual fixtures by demonstration.

Section 5.2 presents related work; it defines the notions *semi-autonomous control* and *shared control*, and puts forward two forms of shared control. The related work is categorized accordingly. Section 5.3 details our approach to learn shared controllers by demonstration. In Section 5.4, the approach is evaluated on a maintenance scenario originating from the Large Hadron Collider (LHC) of CERN, Switzerland. Finally, the method and results are discussed in Section 5.5.

5.2. RELATED WORK

Depending on the implementation, virtual fixtures can either be seen as a form of *semi-autonomous control* or *shared control*. In this chapter, we distinguish the two by the way human and automation control a system. We define a control system to be semi-autonomous when the control of the state variables is *separated*. For example, consider a task that requires the control of both position and orientation of the robot end-effector. This task is performed semi-autonomously when the human controls the position manually, while the orientation is controlled by the assistive system. On the other hand, in *shared control* all state variables are *jointly* controlled by the human and assistive system. The weighting of the control inputs determines the level of automation. Shared control provides a continuum from manual to automated control.

The control intentions of human and automation can be combined at different levels. The majority of virtual fixtures can be seen as a form of Haptic Shared Control (HSC) [133, 135], where the intentions are mixed at the operator interface through haptic interaction, see for example [131,132,136,137]. The haptic communication between operator and assistive system is appealing because it makes the human operator fully aware of the intentions and output of the control system. Furthermore, it relies on human proprioception, which, unlike the visual or auditory senses, is rarely occluded.

Alternatively, the intentions of the operator and assistive system can be mixed after the interface, at the state level. In this case, states given by the operator and the assistive system are fused through a weighted combination. We call this form of shared control State Shared Control (SSC), but other names are used throughout literature: for example, input-mixing shared control [138], mixed initiative control [139], or input blending control [140]. As SSC does not require physical interaction with the operator, it leaves room for vision-based user-interfaces.

Learning of virtual fixtures is a topic of active research. Recently, Raiola *et al.* [137] demonstrated a shared control approach that learns virtual fixtures from data. The user can add new fixtures to the system in order to adapt to new manipulation examples. Bodenstedt *et al.* [141] presented a semi-autonomous system which controls the end-effector orientation autonomously, while the operator controls its translation manually. The assistive behavior is programmed by demonstration. Havoutis *et al.* [142] presented an SSC approach that merges the movements generated by a task model and a human operator. In this approach, the task model is programmed by demonstration. Abi-Farraj *et al.* [143] presented a semi-autonomous approach which encodes tasks as trajectory distributions. These distributions are iteratively refined through collaborative task executions. Pérez-del-Pulgar *et al.* [131] proposed a method to learn HSC by demonstration and assess their method on a peg-in-hole task. During a demonstration phase they record the position of the end-effector and the interaction forces and torques. During reproduction, they infer the desired position from the measured interaction forces and torques using Gaussian Mixture Regression (GMR).

Our work is similar to Havoutis *et al.* [142]. We advance it in two directions. First, our proposed approach relies on the Riemannian framework presented in previous chapters. This allows the consideration of generic rotation in 3D-space. Second, we present approaches to learn both SSC and HSC strategies, while Havoutis *et al.* [142] only considered SSC.

5.3. Programming & Synthesizing Shared Control

In shared control, system state is jointly controlled by a human and an assistive system. The amount to which they can influence the system state defines the *level of automation* [135]— the degree to which a system is automated. The lowest level of assistance yields manual control, while the highest level of assistance results in a fully automated system.

We propose a system that relates the level of automation to the confidence of its inputs. This is achieved by taking into account the confidence level when combining the inputs of the shared control system. When both human and assistance are equally confident, their inputs will be weighted equally. Likewise, when one of the inputs has higher confidence, it will be weighted more heavily.

In our system confidence is expressed by positive-definite matrices. This allows the assignment of confidence on individual state-variables, as well as confidence coupling among state variables. By combining confidence with a desired state, the intentions of both human and automation can be represented by a Gaussian, namely:

$$\mathcal{N}_H = \mathcal{N}_H(\boldsymbol{\mu}_H, \boldsymbol{\Sigma}_H), \tag{5.1}$$

$$\mathcal{N}_A = \mathcal{N}_A(\boldsymbol{\mu}_A, \boldsymbol{\Sigma}_A). \tag{5.2}$$

Here, the center $\boldsymbol{\mu}$ of the Gaussian represents the desired state, and its precision $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$ provides a measure of confidence. The internal model of the assistive system is represented by a Gaussian Mixture Model (GMM), which we denote the task model. The desired state, with a corresponding confidence level, $\mathcal{N}_A$, is thus obtained straightforwardly using GMR. The resulting Gaussian has a full covariance matrix encoding both variance and correlation among the state variables. The parameters of $\mathcal{N}_H$ are set manually, as will be discussed later in this section. Because $\mathcal{N}_A$ is computed online, the confidence of the assistive system can change at each time step. As a result, the level of automation is continuously re-evaluated. This creates a shared control system that shifts control authority online depending on the confidence of the user and automation.

The remainder of this section describes the proposed method in detail. Section 5.3.1 discusses how the assistive system can be programmed by demonstration. Then, Section 5.3.2 describes two different methods in which the intentions of human and assistive system can be combined. Finally, we discuss different structures for the confidence representation in Section 5.3.3.

5.3.1. PROGRAMMING SKILL MODELS BY DEMONSTRATION

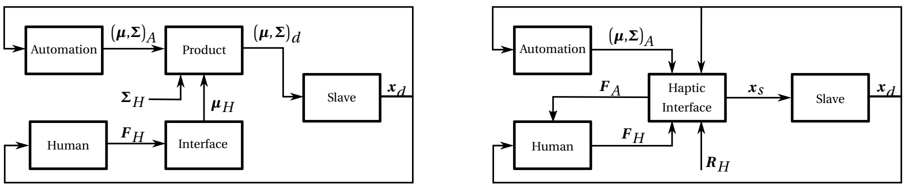
Instead of pre-defining the behavior of the assistive system, we program it by demonstration. We collect a set of example motions while the operator performs the task in question in a direct teleoperation manner. In most teleoperation scenarios, we would be interested in the position and orientation of the robot end-effector. In this scenario, a dataset required to learn a skill would consist of N end-effector poses $\boldsymbol{x}$.

Based on the example motions, we learn a Riemannian GMM using the method described in Section 2.5.6. This model reflects motion data that are identical, recurring and accurately performed as Gaussians with low variance. These Gaussians capture the invariant behavior of the demonstrations—a feature that we exploit during online feedback/motion generation.

We want to ensure that the assistive system only provides assistance in areas where we have learned the skill in question. This is achieved by incorporating a prior into the model which yields non-assistive behavior. This prior is represented as a Gaussian with large variance and added to the GMM. In practice, this Gaussian will be 'activated' when the user enters areas where no skill has been demonstrated. The large variance makes HSC fully compliant, and ensures that SSC ignores the state desired by the assistive system.

Since the skill is modeled as a joint probability density function (pdf) $\mathcal{P}\left(\boldsymbol{x}^{\mathcal{O}}, \boldsymbol{x}^{\mathcal{I}}\right)$ represented as a GMM, we can perform statistical inference using GMR. This estimation yields a Gaussian distribution; e.g $\mathcal{N}\left(\boldsymbol{\mu}^{\mathcal{O}|\mathcal{I}}, \boldsymbol{\Sigma}^{\mathcal{O}|\mathcal{I}}\right)$ parameterized by an expected state $\boldsymbol{\mu}^{\mathcal{O}|\mathcal{I}}$, and covariance $\boldsymbol{\Sigma}^{\mathcal{O}|\mathcal{I}}$ ($\mathcal{I}$ and $\mathcal{O}$ correspond to the input and output variables, respectively).

5.3.2. SHARED CONTROL STRATEGIES



(a) State Shared Control (SSC) (b) Haptic Shared Control (HSC)

Figure 5.1: Block diagrams visualizing the shared control strategies considered in this work.

To achieve shared control, we need to define how the control intentions of the user and the assistive system are combined. We consider two different ways, namely, SSC and HSC. The block diagrams displayed in Figure 5.1 illustrate the different methods. The methods are distinguished by the level at which the inputs of the actors are combined.

We first discuss our implementation of each approach separately, and then highlight their differences in more detail.

STATE SHARED CONTROL (SSC)

SSC combines the inputs after the interface on which the human operates. As depicted in Figure 5.1a, both human and automation generate a control input with a confidence level. Both inputs are represented by a Gaussian: $\mathcal{N}_A$ and $\mathcal{N}_H$. Since both Gaussians encode state variables, we can combine them using Gaussian product: $\mathcal{N}_d = \mathcal{N}_A\mathcal{N}_H$. The Gaussian product produces a weighted average of the centers of $\mathcal{N}_A$ and $\mathcal{N}_H$ that takes into account the variance of the individual variables and correlation among the variables. In effect, this fuses the state issued by the operator with the state predicted by the automation. The product takes into account the confidence of the actors, as this is expressed in the covariance matrices.

HAPTIC SHARED CONTROL (HSC)

HSC combines the inputs at the user interface, as depicted in Figure 5.1b. The input of the agent is conveyed to the human agent through forces applied at the user interface. The operator can either comply or resist these forces. This effectively establishes a shared control system.

We propose to generate the haptic forces or torques using Linear Quadratic Regulator (LQR). This is achieved using the cost function

$$c = \int (\boldsymbol{x}_s - \boldsymbol{\mu}_A)^\top \boldsymbol{\Sigma}_A^{-1} (\boldsymbol{x}_s - \boldsymbol{\mu}_A) + \boldsymbol{u}_A^\top \boldsymbol{R} \boldsymbol{u}_A \tag{5.3}$$

with $\boldsymbol{x}_s$ the current state of the control interface, $\boldsymbol{u}$ the control input applied to the haptic interface, and $\boldsymbol{R}$ a control cost that corresponds to the confidence of the operator. Solving this optimal control problem yields a gain matrix $\boldsymbol{L}$, that is used in the actuation command of the haptic interface

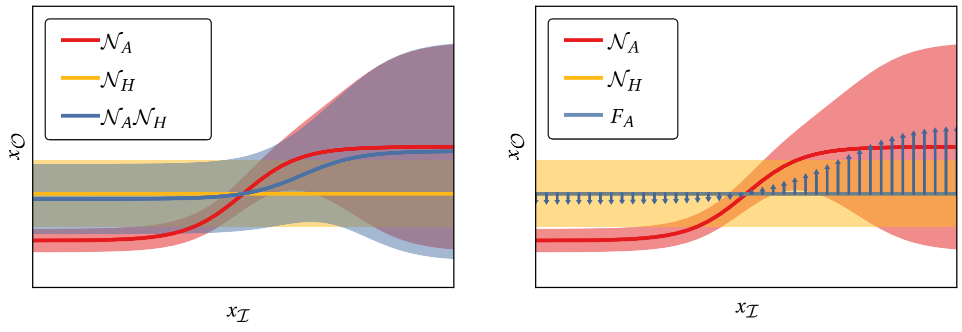
$$\boldsymbol{u}_A = -\boldsymbol{L}(\boldsymbol{\mu}_A - \boldsymbol{x}_s). \tag{5.4}$$

This actuation is felt by the human operator as $\boldsymbol{F}_A$. By displaying the input of the assistive system at the control interface, HSC achieves a high automation awareness. Furthermore, the operator is fully aware of the outcome of the input mixing, because it corresponds to the current state of the haptic interface.

The level of automation is regulated through the magnitude of the applied forces. When the applied forces are such that the human cannot resist them, the system acts autonomously. Manual operation, on the other hand, is achieved when the operator cannot detect the forces. This results in a continuum, and varying levels of autonomy, based on the confidence of the operator's and automation predictions.

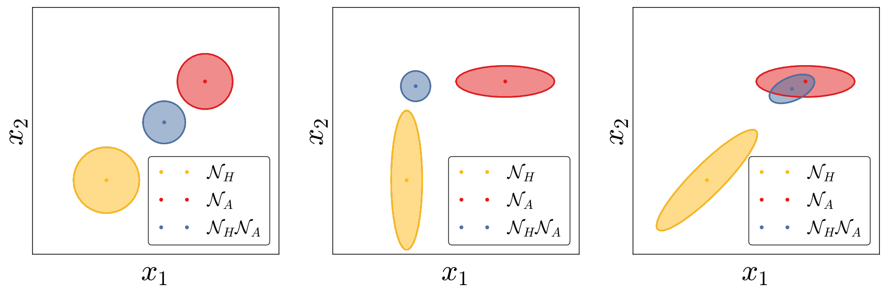
ILLUSTRATIVE COMPARISON

We illustrate the differences between SSC and HSC using Figure 5.2. The graphs show that the output values of the shared control systems (thick blue line) differ. In SSC, the output of the shared control system is the trade-off between the user and assistive input. In contrast, the output of HSC corresponds to the input of the user. Here, we assumed



(a) State Shared Control (SSC) (b) Haptic Shared Control (HSC)

Figure 5.2: Illustration of the proposed shared control approaches on the variable $\boldsymbol{x}_\mathcal{O}$. The input and confidence of the human (yellow) and assistive system (red) are visualized by the lines and the shaded areas, respectively. The outcome of the shared control method is visualized in blue. See Section 5.3 for further description.



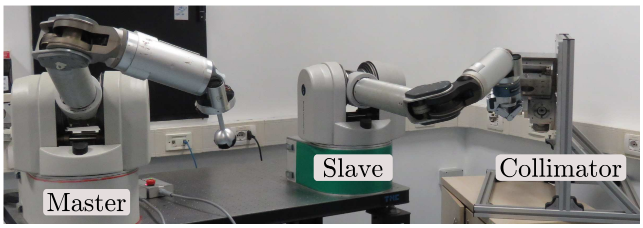
(a) Isotropic (b) Diagonal (c) Full

Figure 5.3: Input mixing using different covariance structures (a–b). The control input of Human operator (H) and autonomous agent (A) are visualized by the red and blue Gaussians (ellipsoids), respectively. The mixed control input is visualized by the green Gaussian. The ellipsoid center indicates the desired state, and its contour the covariance.

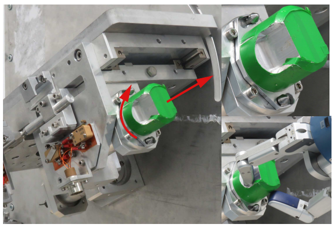
the user counters the force generated by the haptic system (indicated by the blue arrows). HSC thus maintains a direct relation between the state of the interface and the state of the slave robot, while in SSC the state of the interface is not guaranteed to reflect the state of the slave robot. Furthermore, the product of Gaussians used in SSC provides a confidence measure on the output. This measure can be used to determine the stiffness and potential synergies of the slave's end-effector, as discussed in Chapter 3. Such information is not available for HSC.

5.3.3. Confidence structures

The covariance matrix supports a rich expression of confidence. Yet, this support comes at a cost: the amount of parameters to specify. By constraining the covariance structure,



(a) Setup



(b) Task

Figure 5.4: Visualization of experimental setup. Refer to Section 5.4 for a detailed description.

the number of parameters that needs to be specified can be reduced.

Figure 5.3 illustrates how a variety of mixing behaviors can be achieved using different covariance structures. Isotropic Gaussians ($\boldsymbol{\Sigma} = \beta \boldsymbol{I}_d$) put an equal weight on all control variables, but only require specifying one parameter. Diagonal covariance matrices ($\boldsymbol{\Sigma} = \text{diag}(\beta_1, \ldots, \beta_d)$) allow separate weighting on the individual dimensions using d parameters. Full covariance matrices allow the handling of coupling among variables, but require defining $d(d+1)/2$ parameters. Alternatively, the number of parameters can be reduced using subspace clustering as demonstrated by Tanwani *et al.* [144]. Note that the illustrations are based on SSC (product of Gaussians). In HSC, the structure of the control cost matrix $\boldsymbol{R}$ can be used to represent the operator's confidence, as it influences the overall stiffness of the HSC system.

In our experiments we use the demonstration data to estimate full covariance matrices for the automation, but manually define the operator confidence using a diagonal covariance matrix. This keeps the number of open parameters low.

5.4. Experimental Evaluation

5.4.1. Scenario

The LHC at CERN in Switzerland is a hazardous environment. Radiation inside the LHC poses a health threat to maintenance personnel. Currently, maintenance schedules include a period to allow radiation to decay and create a safe working environment. Teleoperation can reduce downtime due to maintenance, as it removes the decay period from the maintenance schedule.

The experimental task is part of the maintenance procedure of a *collimator*—a device used to parallelize particle beams. The LHC can contain up to 152 collimators, which are located in radioactive areas [145]. This maintenance procedure involves the removal and replacement of a protection cover. Removal of the cover is achieved by sequentially moving towards it, grasping it, unlocking it using a 10 degree clockwise rotation, and sliding it from the locking pins (see Figure 5.4b).

5.4.2. Experimental setup

The experimental setup, visualized in Figure 5.4, consists of two Barrett WAM 7-DOF robots and a 1:1 mock-up of the collimator. The left WAM acts as the master device, and is equipped with a haptic ball that allows the teleoperator to control the end-effector pose of the slave. The WAM pictured on the right acts as the slave and is equipped with a three fingered hand (Barrett BH8-280). The hand is programmed to have two states (pre-grasp, and grasp) which are activated by the operator. The mock-up of the collimator contains all parts required for this particular maintenance scenario.

5.4.3. Experimental procedure

The virtual fixture is programmed through kinesthetic teaching; we collect data while manually moving the robot to perform the maintenance operation. Although kinesthetic teaching cannot be applied on site, it can be used in a safe environment prior to the execution of the teleoperation task. Alternatively, one could use demonstration data of an expert teleoperator.

We demonstrated a number of successful removal and replacing attempts of the cap. For each demonstration we recorded position and orientation of the robot end-effector and the collimator. To allow the transfer of the demonstrated skill to different poses of the collimator, we project the recorded end-effector poses in the collimator frame, i.e. the collimator pose represents the task context. The projected data are encoded in a Gaussian $\mathcal{N}(\boldsymbol{\mu}^{\text{skill}}, \boldsymbol{\Sigma}^{\text{skill}})$, with $\boldsymbol{\mu}^{\text{skill}} \in \mathbb{R}^3 \times \mathcal{S}^3$ and $\boldsymbol{\Sigma}^{\text{skill}} \in SPD(6)$. In addition to the trained behavior, we define a variant Gaussian $\mathcal{N}(\boldsymbol{\mu}^{\text{var}}, \boldsymbol{\Sigma}^{\text{var}})$ with relatively large covariance ($\boldsymbol{\Sigma}^{\text{var}} = 10 \cdot \boldsymbol{I}_6$), at the origin of collimator base. This Gaussian ensures that, outside the area of demonstrations, HSC is fully compliant and SSC follows the intentions of the operator. The skill and variant Gaussians are combined in a GMM with equal prior $\pi_i = 0.5$.

The experiment evaluates 3 different control conditions: (1) Manual control (MAN), (2) HSC, and (3) SSC. The shared control methods are used to assist the teleoperator in orienting the end-effector. The desired state of the assistive system is obtained by computing the conditional probability $\mathcal{P}\left(\boldsymbol{q}|\boldsymbol{x}\right)$; a distribution of the desired orientation $\boldsymbol{q}$ given an end-effector position $\boldsymbol{x}$. For SSC we set $\boldsymbol{\Sigma}_H = \text{diag}(0.5, 0.5, 0.001)$. In effect, these settings provide the operator with full control on the rotational axis required for the (un)locking motion of the task. On the other axes the autonomous system has higher confidence. Outside the task area the operator has full control over the position and orientation of the robot. For HSC we selected the control cost matrix $\boldsymbol{R} = \text{diag}(75, 75, 150)$, resulting into approximately equal resistance among all rotational axes.

The 3 control conditions are tested for two different locations of the collimator ($P1$ and $P2$). By changing the location of the collimator, the movements required to remove the cap change significantly. In total 6 trials are performed by each subject (3 control conditions, 2 collimator positions).

We recruited 11 healthy subjects, aged 22–34, that had no prior experience in teleoperation. During the teleoperation the subjects could visually observe the slave robot and the collimator. This is expected to give the subjects better situational awareness, compared to observing the collimator through a 2D vision system, as traditionally used in teleoperation. Yet, part of the scene is still occluded by the hand and arm of the slave. Each subject was shown how to perform the task using teleoperation and was allotted to train the removal and replacement of the cap using all control conditions. The training phase is conducted on location $P0$. Despite the training prior to the experiment, we

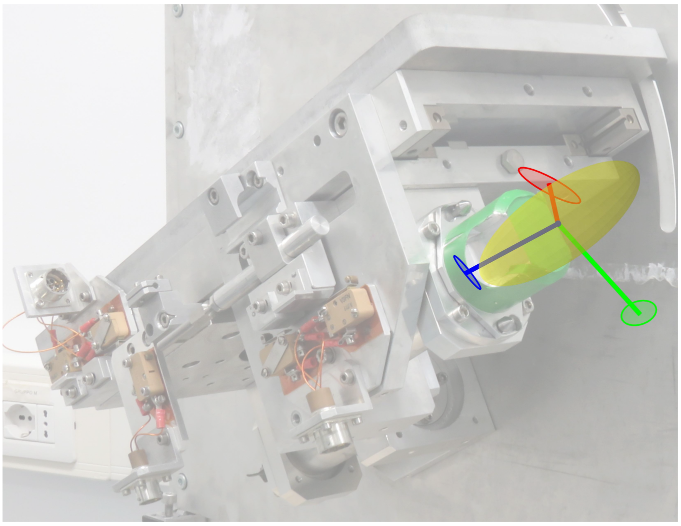


Figure 5.5: Visualization of the trained skill on the collimator. The bound of the yellow ellipsoid indicates one standard deviation of the position covariance. The ellipsoid center indicates the mean. The orthogonal colored lines indicate the mean orientation. The colored ellipsoids at the end of each axis indicate 4 standard deviation of the rotational covariance.

expected subjects to improve their performance throughout the experiment. To avoid such skill improvement to delude the experimental outcome, we randomized the order of the experimental conditions within trials. An example of the order of task executions is: (MAN, HSC, SSC), (HSC, MAN, SSC). Before each trial, the subjects were informed about the type of assistance they would receive.

During each trial we record position and orientation of master, slave and collimator, the state of the hand (grasped/released) and the trial duration. In addition, we asked each subject to fill out a questionnaire about their experience with the proposed shared controllers.

5.4.4. Results

Probabilistic Skill Model

We demonstrated the task on the collimator using kinesthetic teaching (see Figure 5.4b). The obtained model is visualized in Figure 5.5. The figure shows the behavior demonstrated around the cap. The small rotational covariance indicates a fixed orientation of the end-effector, and the positional covariance indicates the desired direction of motion.

During the reproduction phase we generate orientation assistance for the teleoperator. This is achieved by estimating the desired orientation $\boldsymbol{q}$ based on the current position $\boldsymbol{x}$ of the slave end-effector; i.e. $\mathcal{P}\left(\boldsymbol{q}|\boldsymbol{x}\right) \sim \mathcal{N}\left(\boldsymbol{\mu}^{q|x}, \boldsymbol{\Sigma}^{q|x}\right)$. This distribution forms the input of the assistive system.

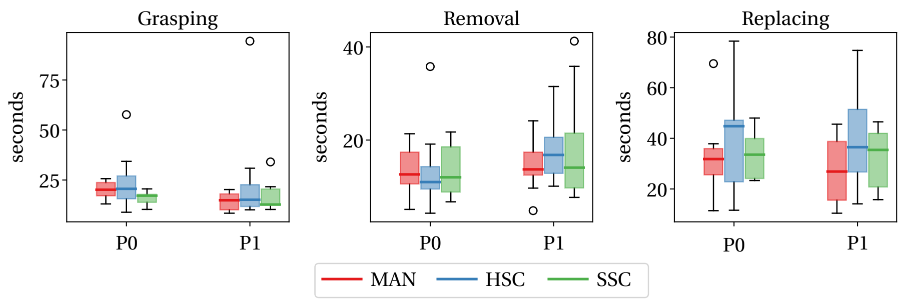


Figure 5.6: Summary of the phase durations for the three teleoperation conditions, evaluated at two locations of the collimator.

	Grasping	Removal	Replacing
MAN ≠ HSC	0.62, 0.21	0.47, 0.16	0.69, 0.24
MAN ≠ SSC	0.06, 0.25	0.64, 0.40	0.63, 0.75

Table 5.1: Overview of the paired t-test results. Null hypothesis: Phase durations of manual teleoperation and the shared control strategy are the same. The entries of the table display the significance level for the two locations of the collimator (P1, P2).

Summary of Experimental Results

We assess the quality of assistance given by the shared controllers based on duration of the different phases in the task. We distinguish three phases in the cap task, namely Grasping, Removal and Replacing. The duration of each of these phases is computed based on the position and grasping signals of the hand. The position signal allows us to determine the location of the end-effector with respect to the collimator. We define a sphere of radius 0.08[m] around the cap which we call the manipulation zone. The grasping time is defined as the duration between entering the manipulation zone and activating the grasp. The removal time is defined as the duration between issuing the grasp command and exiting the manipulation zone. Finally, the replacement time is defined as the duration between entering the manipulation zone for the second time and releasing the grasp.

The measured durations are summarized in Figure 5.6. The results do not display clear differences between the control strategies or positions. To ensure this observation is correct, we performed a paired t-test comparing the results of manual teleoperation with each of the shared control strategies. These results are listed in Table 5.1, and demonstrate that there exist no significant differences ($p < 0.05$) in phase duration among the control methods.

Table 5.2 lists the results gathered from the questionnaire. The subjects were asked to answer each question by selecting one out of five options, which ranged from 'absolutely not' to 'yes, absolutely'. These answers were linearly transformed into the range $[-2, 2]$. Furthermore, we asked the subjects to indicate which form of the teleoperation they preferred: MAN (18%), HSC (27%) and SSC (55%).

		Question: Did you...	Mean	Std
HSC	A	find the provided guiding forces useful?	0.00	1.28
	B	had to 'fight' the provided assistance?	0.55	0.89
	C	feel in control while being assisted by forces?	0.36	0.88
SSC	A	find the provided orientation correction useful?	0.82	0.57
	B	had to 'fight' the provided assistance?	−0.73	1.35
	C	feel in control while being assisted?	1.09	1.08

Table 5.2: Outcome of subjective evaluation. See Section 5.4.4 for discussion of the results.

5.5. DISCUSSION

The presented work shares similarities with the approach of Raiola *et al.* [137]. Nonetheless, our methods are different. Namely, with both SSC and HSC, we perform Gaussian regression from position to orientation while in Raiola *et al.* [137], a virtual fixture is activated based on a notion of closeness.

We demonstrated the implementation of our approach in a real-world scenario. Both HSC and SSC can be trained on demonstrations, and provide a varying level of automation. Although most of the subjects indicated that they prefer a form of shared control while performing teleoperation, our quantitative evaluation does not show that either HSC or SSC increases the teleoperation performance for our chosen measure. As this finding conflicts with previous work on shared control [130, 132, 135], we discuss potential sources that could have caused this contradiction.

The (dis)mounting of the collimator cap was found difficult by our subjects. We found that users experienced difficulty removing and placing the cap over the two lock pins. Although our assistive system attempts to ease these actions by guiding the end-effector orientation, additional fine manipulation is still required to perform the actions. Moreover, if the cap collides with the environment during the replacement, it can slip within the hand. This makes replacement of the cap more difficult, as the orientation desired by the model mismatches the one required for replacement. Both of these issues can be removed by altering the environment, but such modifications would make the application less realistic.

Furthermore, a different input device can be considered. The WAM provides the user with a one-to-one mapping between the master and the slave. However, moving it physically might be too bulky for our subjects. A smaller device, for example the Omega6 from Force Dimension, that is traditionally used for haptic interaction might be more intuitive and easier to use in future experiments.

In our current implementation, we heuristically set the operator confidence for HSC and SSC. These values influence the level of automation that the subjects experience. Using the current settings the subjects indicated they had to fight the forces provided by HSC (see Table 5.2 HSC.B). While the subjects disagreed this was the case for SSC (see Table 5.2 SSC.B). This hint the desired level of assistance was not optimal, and could vary among users. Further research in this direction would be required to confirm these findings.

6

Conclusion & Future Work

We presented a Riemannian approach to PbD. It is founded on three properties that are desirable for skill representations: *geometry-aware*, the ability to combine demonstration data defined on a variety of manifolds; *synergetic*, the ability to extract and synthesize functional coupling from the demonstration data; and *context-adaptive*, the ability to model, synthesize and recognize contexts from demonstration data. As existing skill representations cover these properties only in part, our aim was to develop one that meets all desirable properties. This concluding chapter highlights to what extent Riemannian approach reached our aim, and discusses future research directions.

6.1. Summary of Findings

The base of the proposed skill representation lies at GMM-GMR and TP-GMM. These Euclidean models have shown to be successful in PbD and, for Euclidean data, provide synergetic and context-adaptive properties. The Riemannian generalization maintains these properties and adds geometric awareness.

6.1.1. Geometry-Aware

Geometry awareness lies at the core of the Riemannian approach. In order to make GMM-GMR and TP-GMM geometry-aware, we proposed to reformulate them using the Riemannian Gaussian—the Riemannian analogue of the Euclidean Gaussian [47]. Chapter 2 demonstrated that the Riemannian Gaussian bears all operations required by GMM-GMR and TP-GMM, namely Maximum Likelihood Estimate (MLE), Gaussian conditioning, Gaussian product and linear transformation. The former three operations require a likelihood maximization, which can be solved using an iterative Gauss-Newton algorithm. The potentially non-linear nature of a Riemannian manifold, makes the need for an iterative scheme unavoidable for a generic approach. In our experiments optimization typically took 2-5 iterations to converge.

Noteworthy, the notion of *parallel transport* was shown to be essential for generalization of Gaussian conditioning and Gaussian product. For example, without parallel transport the conditioning output does not follow a geodesic (the generalization of a straight line), and thus not preserves the 'linear' characteristic of Gaussian conditioning.

Unlike Euclidean space, data defined on Riemannian manifolds can have multiple mean points: Riemannian centers of mass. Given a set of demonstrations, the Gauss-Newton algorithm will converge to the closest (local) optimum of the corresponding likelihood function. Only when the data are sufficiently concentrated (contained in a geodesic ball), the algorithm is guaranteed to converge to the global optimum of the likelihood function. Fortunately, given the geometry of the manifold, the potential existence of multiple optima can be determined from the data beforehand. When using unit quaternions to represent orientation data, the existence of local optima is unlikely because SO(3) is almost fully contained in a geodesic ball on $\mathcal{S}^3$.[1]

Practically, the Riemannian approach requires a manifold to be defined in terms of an exponential map, a logarithmic map, and parallel transport. Once these maps are defined, the presented approach is applicable. Moreover, as the Cartesian product of Riemannian manifolds is again a Riemannian manifold, applicability extends to any combination of Riemannian manifolds. In other words, the tool-set required to encode and synthesize a spatial motion in 2D, is no different from the one used to encode a bi-manual task which includes orientation and position in 3D. This is demonstrated by applying the Riemannian framework on $\mathcal{S}^2$ for illustrative purposes, on $\mathcal{S}^3 \times \mathbb{R}^3$ for end-effector manipulation, and on $\mathcal{S}^3 \times \mathbb{R}^3 \times \mathcal{S}^3 \times \mathbb{R}^3$ for bi-manual coordination. Furthermore, Rozo *et al.* [49] and Jaquier *et al.* [50] extended the presented approach te Symmetric Positive Definite (SPD) data.

6.1.2. Synergetic

The covariance structure of the Riemannian Gaussian encodes both variance and correlation between the manifold dimensions. These quantitative connections can be interpreted as functional relations among state variable: synergies. Consequently, the (Riemannian) Gaussian captures potential synergies that occur in demonstration data.

Chapter 3 showed how these synergies can be enabled in state-feedback control strategies. The approach relies on standard LQR, but relates the LQR cost function to the covariance matrix of the Riemannian Gaussian. This step introduces the observed synergies in the control objective. As LQR is not directly applicable on the manifold, we defined it in the linear tangent space at the mean of the Riemannian Gaussian.

The LQR-based controllers act as virtual spring-damper systems that are spanned across the manifold dimensions. The synergies make their perturbation response mimic the coordination patterns observed in the demonstration data. In analogy to human-motor control, this behavior can be perceived as reflexive: attempting to maintain functional integrity. Practically, the state-feedback controllers achieve this response without the need for additional, (potentially) expensive regression steps.

The nature of the rotational manifold SO(3) prevents the existence of a globally stable continuous state-feedback controller. This holds for both SO(3) and its parameterizations (e.g. unit quaternions). Despite this intrinsic limitation, state-feedback controllers on SO(3) function well because the attraction domain of the (unstable) equilibrium is nowhere dense.

[1] A geodesic ball on $\mathcal{S}^3$ covers a hemisphere with exclusion of its boundary which corresponds 2π rotation.

As a natural trade-off between spatial and rotational length-scales does not exist, it needs to be specified manually for controllers that jointly control position and orientation. In our approach this trade-off is made through the (diagonal) control cost matrix. In our experiments a single control cost was used to achieve different synergetic controllers. This indicates that control cost is more robot dependent than task dependent.

6.1.3. Context-Adaptive

Task-Parameterized Gaussian Mixture Model (TP-GMM) is a skill representation that encodes data in multiple local coordinate systems, each representing the skill from a different perspective. By contextualizing the local perspectives in a common coordinate system and fusing their skill representations, generalization to new context is achieved.

Chapter 4 generalized TP-GMM to Riemannian manifolds using the Gaussian operations presented in Chapter 2. Additionally, it introduced a method for context inference; a way to deduce context for known tasks from observations. These extensions permit TP-GMM to perform modeling, synthesis and inference for context-adaptive tasks that involve position and orientation data.

We presented an overview of the types of context parameterizations that are provided by (Riemannian) TP-GMM. The most generic context parameterization involves affine transformations. These allow scaling, translation and rotation of individual coordinate systems. Yet, the flexibility provided by the affine group implies a large number of context parameters. This might be challenging for its future application in context inference. Alternatively, sub-groups of the affine group, such as the rigid-body transformations, provide a lower dimensional alternative that is sufficient for many scenarios.

The generalization capabilities of the context-adaptive approach have been evaluated in simulation for synthesis and inference. The evaluation assessed this capability based on context-likelihood, a (Gaussian) measure that judges the chances that a context was observed before. Based on this measure, we found that that TP-GMM is able to generalize properly for likely context ($\sigma < 1$), but generalization capabilities decay for less likely context. In these cases, the trade-off achieved by product of Gaussians puts too much weight on irrelevant motion aspects. As a result, synthesis still mimics the demonstration data, but does not guarantee task constraints are fully met. Yet, using the context-likelihood, the robot can assess its chances of success under a given context in advance. This ability could serve as safety guard in critical scenarios, and prevent the robot from damaging its environment.

Furthermore, we evaluated Riemannian TP-GMM in a Pick & Place task, and a box-tape task, which involved both position and orientation information in 3D space. The box-tape task displayed the context-adaptive competence of TP-GMM under affine transformations. It allowed the model to adapt to boxes of known and unknown shape that were placed in different poses. The Pick & Place task, executed on the Schunk robot, demonstrated context adaptation for rigid-body transformation. But also revealed that regularization steps might be required during reproduction. The reproductions of this task showed that the model captured correlations which were demonstrated unintentionally. These effects can be attenuated through covariance shrinkage.

The concept of context inference is novel in the light of TP-GMM. This thesis, presented the generic problem of context inference, and proposed an EM-based method to solve it. The optimization problem required for context inference is prone to have local optima. To avoid them, we proposed a regularization method which alters the expectation step of EM. Although effective, the experimental evaluation showed that the proposed method requires additional research to make context inference more reliable. Suggestions to further improve context inference are given below.

6.1.4. Programming Shared Controllers by Demonstration

Chapter 5 presents how the material of chapters 2, 3 and 4 can be used in shared control. Generally, shared control strategies combine inputs of a human operator and an assistive system. In the presented approach, both inputs are represented by a Gaussian; its mean indicates a desired state, and its covariance relates to the confidence of this state. We proposed to program the behavior of the assistive system by demonstration, and present two ways to fuse intentions of human and automation, namely HSC and SSC. Both approaches continuously trade-off their inputs based on their confidence, thereby establishing a constant re-evaluation of control authority.

The control strategies were compared to manual teleoperation in a user-study. Although the objective results do not confirm that the use of SSC or HSC improves task completion time, the subjects preferred shared control over manual control.

6.2. Future Work

The use of geometrical concepts is not new in robotics and the related field of human motor control. In robot control, it has proved to be a valuable tool for control, dynamics, and design algorithms [85–87, 146], and it is also gaining attention in the learning community [48–51, 147]. Geometrical concepts such as *geodesics* and equi-affine velocities can be associated to phenomena found in human motor control [148, 149]. This thesis contributed to this roadmap with a Riemannian view on PbD. In this closing section we explore future work: what opportunities do we see for the Riemannian approach to PbD?

6.2.1. Short Term Outlook

Below is a list of concepts, which we deem practically applicable. These ideas arose during the development of the Riemannian approach, but did not come to light due to time constraints. Some of them have been briefly mentioned within the preceding chapters, and will be elaborated in more detail.

Including robot Dynamics in Control

Our LQR-based approach for synergetic control defines both state and control cost in task space. Although it seems sensible to specify task-related cost in this space, physically, control cost appears at actuation level in joint space. The ideal combination would define task-related cost in task space—independent from robot kinematics and dynamics—and define the control cost at joint level.

The corresponding optimal control problem will be more complex, as it needs to consider the non-linear dynamics and kinematics. In return, the controller truly acts according to the minimal intervention principle [12]; trading off task performance with actuation-based control cost. The resulting policy might demonstrate more energy-efficient postures, exploit the passive dynamics in task execution, or find postures in which intrinsic motor noise interferes least with task performance [13]. As suggested by Park [150], a geometric approach to robot dynamics and control could yield efficient solutions in this direction [86, 146].

Covariance weighted Inverse Kinematics

In this thesis, correlation information was used to ease the definition of controller gains. Covariance might also become of use in inverse kinematics. Efficient inverse kinematics solvers rely on a regularized least-squares solver. The regularization term in this optimization term defines the allotted solution error. As the covariance matrix contains such information, it might be used to establish task-specific inverse kinematics solvers. Such task-specific solutions are useful for applications where only few degrees of freedom are available, or left when multiple tasks are combined.

Active Learning

For extreme context extrapolation, generalization cannot always be reliably achieved. In such cases, important (invariant) task features, are too strongly influenced by the unimportant (variant) features during the fusion step. Active learning could provide a way to resolve this. In an active learning scenario, we could measure the confidence that a context has been observed before. If this likelihood is low, generalization is likely to fail. In that case, the robot can improve its skill by asking the user to demonstrate the proper response for the new context. This approach falls in line with the active learning scheme presented by Maeda *et al.* [123].

Discrete Context Constraints

The generalization capabilities of TP-GMM are currently limited by the variability of the context encountered during the demonstration phase. Extrapolation capacity could be improved by transforming strong motion features in constraints, or by omitting weak features. This can be achieved by modifying the eigenvalues of the covariance matrices of variant and invariant states. For example, if a Gaussian has only large eigenvalues, it is unlikely it contains any important information for motion synthesis, or recognition. To ensure these states do not affect motion synthesis in case of strong generalization (extrapolation), we can give them infinite variance. In effect, this is a discretization step, which locally discards coordinate systems during the fusion step of TP-GMM.

Improved Context-Inference

Chapter 4 presented a novel approach for context inference. Despite the proposed regularization, the evaluation showed context inference is not always successful. This was caused by an improper temporal sequence detection: the state activation order did not match the ground truth. Future work could address this by explicitly modeling the state activation order and duration using a Hidden Markov Model (HMM) [128] or Hidden Semi-Markov Model (HSMM) [151]. Besides alleviating the activation order problem, this could enable context inference from partially observed motions. This ability can be attained through the HMM forward-backward algorithm [128].

In this thesis, the evaluation of context-inference was restricted to rigid-body transformations for two reasons: first, the dimensionality of the inference problem is significantly larger compared to rigid-body pose estimation. As inference was already challenging for rigid-body pose, its use has not been attempted for affine transformation.

Secondly, Riemannian optimization on affine transformations is computationally expensive as no closed-form solution exists for the (matrix) exponential and (matrix) logarithm maps. The former could be attempted for HMM-based modeling, and the latter matter might be resolved using retraction mappings [67]. These mappings approximate the exponential and logarithmic map, and are computationally more efficient.

6.2.2. Long Term Perspectives

We close with two more general topics: we introduce the motivation of a "Riemannian dynamic primitive", and discuss how future work could address "*what-to-imitate?*"

Riemannian Dynamic Primitives

In this work, we separated regression (GMR) and control (LQR). For the Euclidean case, these steps can be merged, as we showed in previous work [19]. This is achieved by modeling the local motion dynamics (position and velocity) in a Gaussian, and the Gaussian activation sequence and sojourn using a HSMM [151]. Reproduction involves the generation of a (discrete) state activation order, which can directly be transformed in control signals using Model Predictive Control (MPC) [152].

Compared to traditional time (or phase)-driven systems, this approach does not require temporal alignment of demonstration data. Yet, it administers temporal constraints, unlike the dynamical system approach (e.g. [35]). The use of HSMM and HMM could form a bridge between low-level trajectory-based approaches and the higher-level symbolic approaches of PbD. The discrete nature of HMM provides the means to symbolize (groups of) local motion dynamics, and couple them in a desirable order. Using MPC, symbol compositions can be merged to ensure smooth reproductions. Extending the Riemannian approach in this direction would yield similar benefits.

Hogan and Sternad propose a broader definition for dynamic primitives [17]. They argue primitive representations should integrate various sources of dynamic information. They observe that despite limitations of the neuromuscular system humans outperform robots in tool-use, a hallmark of human behavior. They hypothesize that humans can only achieve such behavior by encoding their behavior in primitive dynamic actions. These dynamic primitives are composed of atoms, exemplified by discrete or cyclic movements, or impedance behaviors. As such information involves different manifolds, these primitives would require a unified framework to connect the atoms into primitives. We believe the presented Riemannian framework is up to this challenge. It provides a statistical framework that can handle a variety of manifolds encountered in the proposition of Hogan and Sternad. The Riemannian approach can encode discrete, periodic, and impedance behavior in a single model. Discrete time-driven movements by representing the temporal signal on a 1D Euclidean space. Periodic motion by representing the phase signal on the unit-circle $\mathcal{S}^1$. Furthermore, the framework can handle task-space information which includes position ($\mathbb{R}^3$), orientation ($\mathcal{S}^3$ or SO(3)), and impedance information ($\mathcal{S}^+$).

Future work would investigate how our previous work [19] and the dynamic primitives of Hogan and Sternad [17] could be merged with the Riemannian approach. Such an approach is expected to involve (heavy) optimization, yet by exploiting the manifold structure efficient optimization solutions might arise.

WHAT TO IMITATE?

In this thesis, we addressed the question: *What to imitate?*; what features in the demonstration data characterize the objectives of the task. A general approach to answer this is to estimate an *imitation metric* from the data. This metric should somehow capture the imitation objectives. By maximizing this metric the robot can imitate the demonstrated behavior, i.e. determine *how to imitate?*.

This imitation metric can be found in different branches of robot learning: the reward function in (deep) Reinforcement Learning (RL) [10, 118, 153] can be considered as an imitation metric, as it describes optimal behavior; akin RL, the cost functions used in optimal control [154, 155] are metrics of imitation. These links become more clear in the light of inverse RL or Inverse Optimal Control (IOC) [156–161], where demonstration data are used to compose a reward (or cost) function from a set of (parameterized) candidate rewards functions. Both the candidate and composed reward functions are imitation metrics, as they describe optimal behavior.

Methods based on TP-GMM and GMM-GMR represent the more classical approach to PbD. They are typically used to represent time-based motion trajectories. Their likelihood functions represent an imitation metric: the ability of a motion to maximize this metric, reflects its match to the demonstrated data. Furthermore, TP-GMM encodes skills from multiple, competing perspectives, which is reminiscent of the candidate reward functions used in RL and IOC. The ability to cast TP-GMM into an optimal control problem strengthens its parallel with IOC. The difference between our PbD methods and IOC or RL lies in level of detail described in the imitation metric. The metrics used in our method are relatively concrete (low-level), while IOC and RL define more high-level goals. In this perspective, this thesis contributed by generalizing the ability to specify metrics of imitation to Riemannian manifolds, and might also serve IOC by defining Riemannian reward candidates.

The links between IOC and PbD raise a variety of questions: Is the structure of TP-GMM suitable for context-adaptive reward shaping in the field of IOC? Can its structure be used to represent high-level goals more generally? Can a Riemannian approach improve IOC efficiency by constraining the combination of reward functions to a manifold? Can the Riemannian approach serve at a meta-level? i.e. can knowledge about reward landscape geometry be used to make exploration more efficient? More generally, questions arise about the relation between the Riemannian metric and imitation metric: Can PbD, which essentially uncovers an imitation metric, be phrased as metric learning [45, 46]? Can it be related to the discovery of the manifold structure? These, among others, are questions worthy of further investigation.

LIST OF ACRONYMS

DMP	Dynamic Movement Primitives
EM	Expectation Maximization
HMM	Hidden Markov Model
HSMM	Hidden Semi-Markov Model
HSC	Haptic Shared Control
GMM	Gaussian Mixture Model
GMR	Gaussian Mixture Regression
GP	Gaussian Process
GPR	Gaussian Process Regression
IIT	Istituto Italiano di Tecnologia
IOC	Inverse Optimal Control
LHC	Large Hadron Collider
LQR	Linear Quadratic Regulator
MLE	Maximum Likelihood Estimate
MPC	Model Predictive Control
PbD	Programming by Demonstration
pdf	probability density function
ProMP	Probabilistic Movement Primitives
RL	Reinforcement Learning
SEDS	Stable Estimator of Dynamical Systems
SPD	Symmetric Positive Definite
SSC	State Shared Control
TP-GMM	Task-Parameterized Gaussian Mixture Model

A

Algorithms

A.1. Initialization algorithms for EM

Expectation Maximization (EM) is commonly used for parameter estimation of GMM. Within this thesis we use mixtures of Riemannian Gaussian, and present a modified version of the EM algorithm in Section 2.5.6. Like any EM algorithm, this algorithm requires an initial guess of the parameters. Two commonly applied initialization methods are K-means and K-bins.

A.1.1. K-means

K-means is hard-clustering technique that computes K means of a given data set. The means are computed using EM. As K-means does not consider the covariance of the data and uses a hard-assignment, it generally converges faster than EM for GMM [94]. However, like EM for GMM, K-means will converge to the closest local optimum. To increase probability of discovering the global optimum, K-means can be ran several times using different initial conditions. Algorithm A.1 gives pseudo-code for K-means on Riemannian manifolds. First, the K means $\boldsymbol{\mu}_k$ are initialized by equating them to randomly selected unique data points (line 3). Then, the total distance between the cluster centers and the cluster data points minimized. This is done iteratively. The expectation step assigns point each data point to the closest mean in an expectation step (lines 7–10). Then, the maximization step updates the mean of each clustering the assigned data points (lines 7–10). The logarithmic map in lines 10 and 14 enables the generalization of K-means to Riemannian manifolds. The mean $\boldsymbol{\mu}_k$, computed in the M-step, is unique as long as $\boldsymbol{x}_n \forall r_{n,k}$ are contained in a regular geodesic ball (see Section 2.5.1). After convergence, the estimated means can be used to compute Covariance and priors (lines 18 and 19).

A.1.2. K-bins

K-bins is a non-iterative algorithm that divides data in K bins (clusters) according to a Euclidean measure label assigned to each data point. When the data set consists of multiple temporally aligned demonstrations, the time stamps are ideal as labels.

Algorithm A.2 describes an implementation of K-bins. The input of K-bins consists of N sets of a label $b_n \in \mathbb{R}$ and data point $\boldsymbol{x}_n \in \mathcal{M}$, and the number of bins K. First, the label range is determined (lines 2 and 3). This range is split into K equal ranges of length δ (line 4). Then, for each range k, the mean, covariance and prior are computed for all points whose label lies within this range (lines 5–10).

Algorithm A.1 K-means initialization

```
function KMEANS($\boldsymbol{x}_{1:N}$, $K$)
    # Initialization:
    $\boldsymbol{\mu}_k \leftarrow \text{Random}(\boldsymbol{x}_{1:N}, K)$        ▷ Randomly select data points as initial means

    # Likelihood Maximization
    while $r_{n,k}$ change do
        # E-Step:
        for $k \in \{1, \cdots, K\}$ do
            for $n \in \{1, \cdots, N\}$ do
                $r_{n,k} = \begin{cases} 1 & \text{if } k = \arg\min_j \text{Log}_{\boldsymbol{\mu}_j}(\boldsymbol{x}_n)^\top \text{Log}_{\boldsymbol{\mu}_j}(\boldsymbol{x}_n), \\ 0 & \text{otherwise} \end{cases}$

        # M-Step:
        for $k \in \{1, \cdots, K\}$ do
            $\boldsymbol{\mu}_k \leftarrow \arg\min_{\boldsymbol{\mu}_k} \left(\sum_{n=1}^N r_{n,k} \text{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n)^\top \text{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n)\right)$

    # Compute initialization for EM
    for $k \in \{1, \cdots, K\}$ do
        $\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_n^N r_{n,k} \text{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n) \text{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n)^\top$
        $\pi_k = \frac{\sum_{n=1}^N r_{n,k}}{N}$
    return $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k\}_{k=1}^K$
```

Algorithm A.2 K-bins initialization

```
function KBINS($\{b_n, \boldsymbol{x}_n\}_{1:N}$, $K$)
    $b_{\min} = \min(b_{1:N})$
    $b_{\max} = \max(b_{1:N}) + \epsilon$        ▷ $0 < \epsilon \ll 1.0$ ensures all $\boldsymbol{x}_n$ are assigned
    $\delta = (b_{\max} - b_{\min}) / K$

    for $k \in \{1, \cdots, K\}$ do
        $r_n = \begin{cases} 1 & \text{if } \delta(k-1) \leq (b_n - b_{\min}) < \delta k, \\ 0 & \text{otherwise} \end{cases}$
        $\boldsymbol{\mu}_k \leftarrow \arg\max_{\boldsymbol{\mu}_k} \left(-\frac{1}{2} \sum_{n=1}^N r_n \text{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n)^\top \text{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n)\right)$
        $\boldsymbol{\Sigma}_k = \frac{1}{\sum_{n=1}^N r_n} \sum_n^N r_n \text{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n) \text{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n)^\top$
        $\pi_k = \frac{\sum_{n=1}^N r_n}{N}$
    return $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k\}_{k=1}^K$
```

A.2. EXPECTATION MAXIMIZATION

EM is an iterative algorithm to estimate parameters of models with latent—unobserved—variables [94, 162]. In the case of GMM, the unobserved variable is the Gaussian that generated an observation. Assume given the complete data set $\{\boldsymbol{X}, \boldsymbol{Z}\}$ consisting of the observed data $\boldsymbol{X}$ and the latent data $\boldsymbol{Z}$.The parameterized distribution over the full data set is given by the joint distribution distribution $\mathcal{P}(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta})$. Where $\boldsymbol{\theta}$ are the parameters to be estimated. Yet, in reality only the marginal distribution,

$$\mathcal{P}(\boldsymbol{X}|\boldsymbol{\theta}) = \sum_{\boldsymbol{z}} \mathcal{P}(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta}), \tag{A.1}$$

can be observed.

Consider the following decomposition of the observed distribution.

$$\mathcal{P}(\boldsymbol{X}|\boldsymbol{\theta}) = \mathcal{L}\left(q, \boldsymbol{\theta}\right) + \mathrm{KL}\left(q \parallel p\right), \tag{A.2}$$

with

$$\mathcal{L}\left(q, \boldsymbol{\theta}\right) = \sum_{\boldsymbol{z}} q(\boldsymbol{Z}) \ln\{\frac{\mathcal{P}(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta})}{q(\boldsymbol{Z})}\}, \tag{A.3}$$

$$\mathrm{KL}\left(q \parallel p\right) = -\sum_{\boldsymbol{z}} q(\boldsymbol{Z}) \ln\{\frac{\mathcal{P}(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\theta})}{q(\boldsymbol{Z})}\}, \tag{A.4}$$

the lower bound[1] and Kullback-Leibler divergence, respectively. Given some initial estimate of the parameters $\boldsymbol{\theta}^{old}$ EM iteratively performs two steps: First, the lower bound on the likelihood $\ln \mathcal{P}(\boldsymbol{X}|\boldsymbol{\theta})$ is maximized with respect to q. This is achieved by setting q equal to the posterior $\mathcal{P}(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\theta})$. The posterior maximizes the lower bound because it minimizes the Kullback-Leibler divergence (A.4) given $\boldsymbol{\theta}^{old}$. Sequentially, the maximization attempts to maximize $\mathcal{L}\left(q, \boldsymbol{\theta}\right)$ by changing $\boldsymbol{\theta}$ while keeping the previously found $q(\boldsymbol{Z})$ fixed. This step guarantees an increase of $\ln \mathcal{P}(\boldsymbol{X}|\boldsymbol{\theta})$, given that we can find $\boldsymbol{\theta}$ that increases $\mathcal{L}\left(q, \boldsymbol{\theta}\right)$. This can be understood by considering (A.2) and realizing that the Kullback-Leibler was minimized in the E-step and cannot further decrease because $\mathrm{KL}\left(q \parallel p\right) \geq 0$.

[1] Note that $\mathcal{L}\left(q, \boldsymbol{\theta}\right)$ is a lower bound because the Kullback-Leibler divergence is by definition equal or greater than zero.

B

Context-Adaptive Experiments

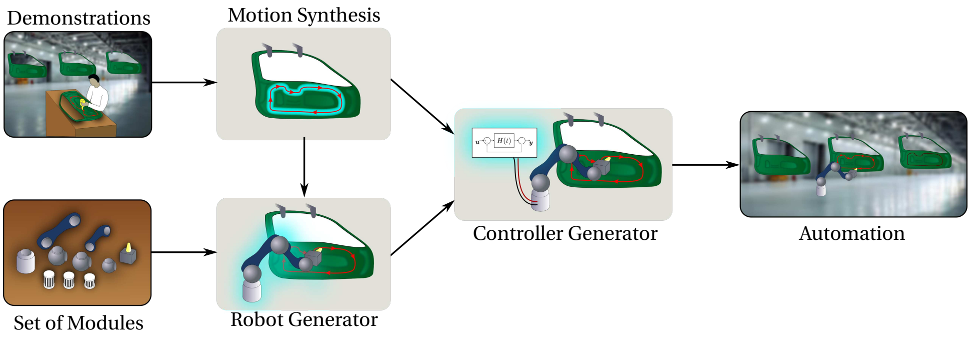


Figure B.1: A modular approach to industrial automation.

In Section 4.5 we describe the results of different experiments. This appendix describes these experiments in more detail.

B.1. A MODULAR APPROACH FOR AUTOMATION

This section describes work done within the SMART-E project [57]. The presented experimental results are obtained in collaboration with Andrea Giusti, Esra Icer and Aaron Pereira of the Technical University of Munich (TUM).

We start by describing the background of the modular approach to automation. Then, we discuss the experimental evaluation of the approach, thereby focusing on the demonstration and synthesis aspects of the Riemannian approach. A video of the two experiments is available online. Throughout the text, hyperlinks are used to highlight specific parts of this video.

B.1.1. BACKGROUND

In classical industrial environments robots perform only a small set of tasks for long periods of time. They are selected because their strength and kinematic structure suited the task requirements, and their behavior is hand-coded by an expert programmer.

The classical approach to automation is less suited for flexible automation, where tasks might change from hour-to-hour or day-to-day. Buying a dedicated robot for each set of tasks is uneconomical and manual coding is too time consuming. Flexible automation requires a robot whose hardware and software are more easily adapted to new tasks.

In this perspective, we investigate a modular approach to industrial automation in which demonstration data drives both robot composition and robot programming. The concept is visualized in Figure B.1 and assumes that a task to automate and a set of modules are given. In order to automate the task, the user demonstrates it a number of times. The demonstrations convey to the system how the task is executed and how it should adapt to context variations. The demonstrations are used to generate a task model which is used in two ways: it is used to find the best composition of modules to meet a given optimization criterion (e.g. energy consumption or execution speed, see

Icer *et al.* [163, 164]); and it is used to replicate the task after robot assembly. To replicate the task, the robot requires a suitable controller. To achieve this, we rely on the work of Giusti *et al.* [165, 166]. They assume each robot module stores its kinematic and dynamic properties. After assembly this information is collected by a central control unit to derive a model-based controller. Compared to decentralized control approaches for modular robots, this approach is appealing as it allows the use of standard model-based controllers. Using the derived controller, the demonstrated task is automated and ready for production.

B.1.2. EXPERIMENTAL SETUP & TASK DESCRIPTION

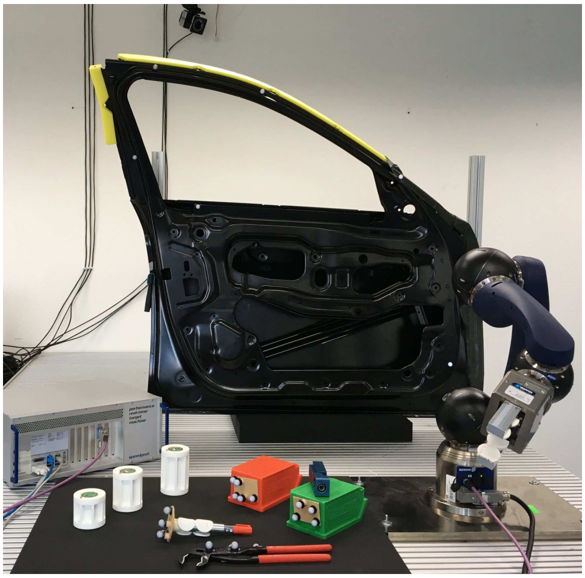
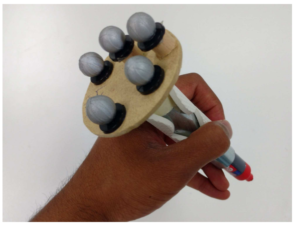


Figure B.2: Overview of the experimental setup.

The experimental setup used to evaluate the modular approach is visualized in Figure B.2. The modular system consists of a Schunk modular robot[1], three additional custom modules, and a Vicon infrared motion tracking system. The modular system is used to perform two different tasks: a trajectory tracking task, and a pick & place task. The former involves the tracking of a profile inside the car door. The latter involves the movement of an object between the green and red boxes.

The two tasks are demonstrated using dedicated demonstrator tools, which are displayed in Figure B.3. Markers are attached to each tool to allow a motion tracking system to recognize and record their motion. Each tool has a pre-defined mapping to a specific end-effector module. The use of dedicated demonstrator tool resolves the correspondence problem [3] in a practical manner, as they explicitly define which information should be recorded.

[1]Although Schunk does not sell the robot components separately, the system can be used in a modular fashion: the modules can be combined and controlled in different configurations.


(a) Pen tool

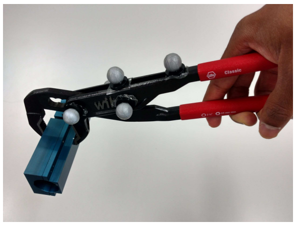

(b) Pick & Place tool

Figure B.3: Demonstrator tools

B.1.3. EXPERIMENTAL RESULTS

DOOR TASK

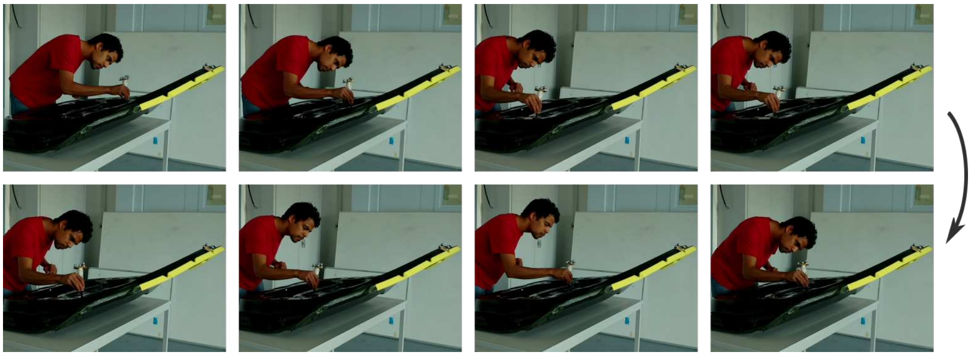


Figure B.4: Snapshots of a typical door task demonstration.

To demonstrate the profile-tracking task, the car door is placed on a table in such a way that the user can easily demonstrate the motion. The objective of the task is to track a profile inside the door with sufficient pressure and correct orientation. In practice, such a motion is required to attach fabric onto the door. In our experiment, only the position and orientation of the pen tool are recorded. The pressure is obtained by compression of a spring that is part of the pen tool. The trajectory consists of both position and orientation information. The orientation information ensures that the pen trajectory is perpendicular to the curved profile.

The task is demonstrated three times. During each demonstration the location of the door and the demonstrator tool are recorded using the tracking system. Snapshots of a typical demonstration are given in Figure B.4 and shown in the video. In this scenario, the context consists of the pose of the door. By encoding the tool trajectory with respect to the car door, we obtain a context-independent task representation. After the demonstration phase, excess data at the start and end of the demonstrations are trimmed using a graphical user interface. Furthermore, the temporal signal is re-scaled to make it range from 0 to 1. These two steps improve the temporal alignment of the demonstration data, thereby contributing to a better model. Finally, the context-independent data (compris-

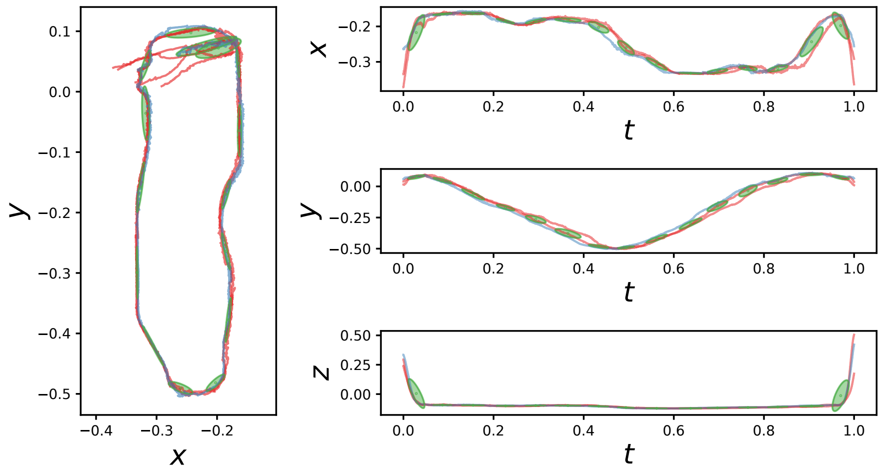


Figure B.5: Illustration of the demonstration data and the task model of the door task. The demonstration data are displayed by the three colored lines, and Gaussians of the GMM are visualized by the three colored ellipsoids.

ing position, orientation and temporal information) are encoded in a Riemannian GMM with 15 states using EM (Section 2.5.6). The number of Gaussians was selected using the Bayesian Information Criterion (BIC) [125]. Figure B.5 shows the spatial representation of the model, together with the GMM obtained after EM.

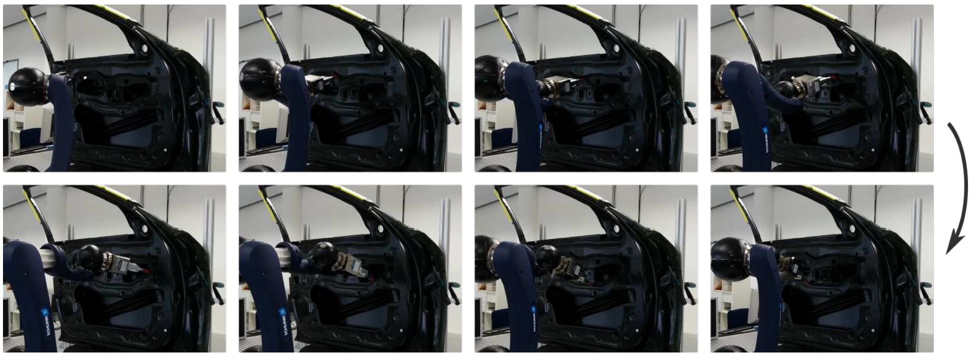


Figure B.6: A typical reproduction of the door task.

After modeling the task, the door was mounted close to the base-module as illustrated in Figure B.2. The task model is contextualized using the new door pose. Using the contextualized model, the task motion is synthesized. This is achieved using GMR with time as input, and position and orientation as output. The synthesis result serves as input of the robot configuration generator, and the reproduction. Here, we only focus on the reproduction part. And assume the optimal configuration is found and assembled.

To keep the joint velocities within the hardware limitations, the generated trajectory is dynamically rescaled in time. This process can be automated, see for example [124].

The reproduction is displayed using snapshots in Figure B.6, and in the video. They show a smooth trajectory which complies with the demonstrations. Both position and orientation aspects of the demonstrations are preserved and properly adapted to the new context (door pose).

Pick & Place task

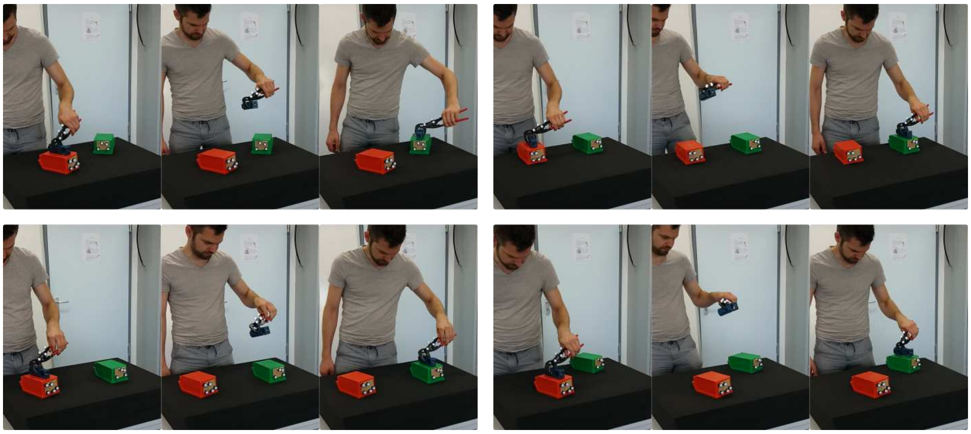


Figure B.7: 4 demonstrations of the Pick & Place task.

The second task considered in the modular framework is a pick & place task. In this task, the context comprises the pick and place locations, each parameterized using a rigid-body pose. By encoding the demonstration data in these local coordinate systems, the task can be generalized to new situations. This has already been we demonstrated in Section 4.4.2 using a 2D peg-in-hole task. In this experiment, we show that the Riemannian approach can achieve generalizations of tasks that include orientation data.

The goal of the pick and place task is to pick an object from the red box, and place in on the green box. 4 of the 9 demonstrations are visualized in Figure B.7, and featured in the video. Each demonstration is recorded in a different context. Each context having different position and orientation of the goal and target box. During the demonstrations we record the box poses, the tool pose and a signal which indicates grasp/release of the gripper. The latter (binary) signal is verbally communicated during the demonstrations. After the demonstration phase, the end and start of each demonstration is trimmed using a graphical user interface, and the temporal signal linearly re-scaled to make it range 0–1. This pre-processing step improves temporal alignment of the demonstration data.

The context-independent data consist of a temporal signal, two pose signals (one for each local coordinate system), and a grasp signal. The data are thus defined on the Riemannian manifold $\mathcal{M}_{pu} = \mathbb{R}^1 \times (\mathbb{R}^3 \times \mathcal{S}^3) \times (\mathbb{R}^3 \times \mathcal{S}^3) \times \mathbb{R}^1$. They are encoded in a Riemannian GMM consisting of 6 states (empirically set). The marginal distribution of position data, and demonstration data are visualized in Figure B.8. The demonstration data reveal distinct variation patterns in the pick and place frames (of reference). The pick frame shows invariance at the start of the motion (approximately $0.1 \leq t \leq .4$), and the

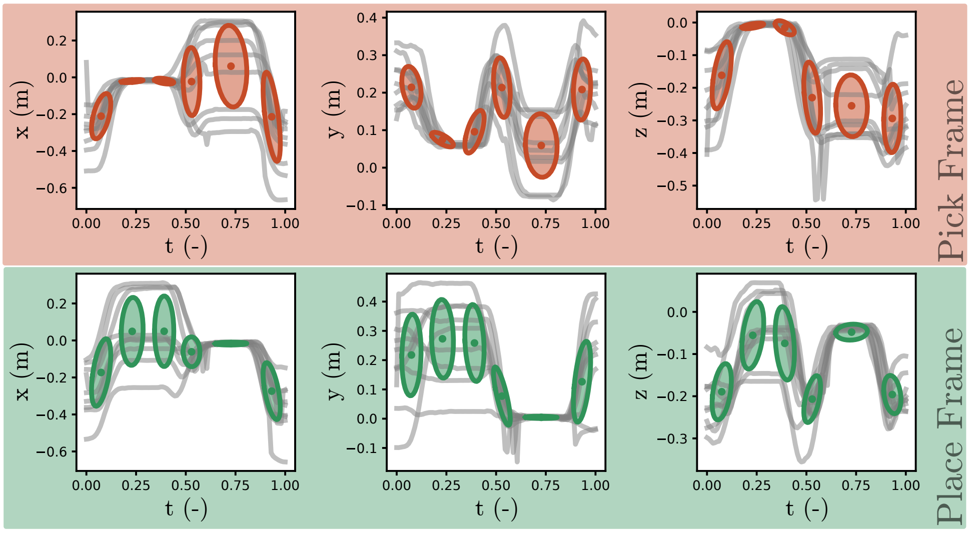


Figure B.8: Context-independent Pick & Place model.

place frame at the end of the motion (approximately $0.6 \leq t \leq .8$). This information is captured in the covariance of the GMM.

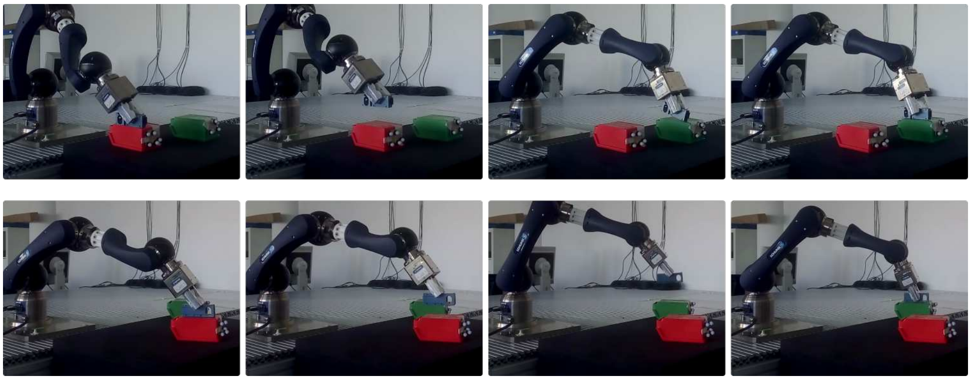


Figure B.9: Snapshot of 2 reproductions of the Pick & Place task.

After assembling the optimal configuration, the motion is reproduced in three unseen contexts. Snapshots of two reproductions are given in Figure B.9, and the three full reproductions are featured in the video. The reproductions demonstrate that the robot can adapt its motion to the new contexts. However, during placement the robot slightly pushes the object onto the place location, a behavior which has not been demonstrated.

We discuss the cause of this behavior and a potential remedy using figures B.10 and B.11. In (a), both figures display the spatial mean outputs of GMR and the Gaussian product using the colored lines. Subplots (b–c), display information about the GMR and product outputs at $t = 0.75$. (b) displays the error between the GMR outcomes and the product outcome(i.e. $\mathrm{Log}_{\boldsymbol{\mu}_{prod}}\left(\boldsymbol{\mu}_{gmr}\right)$), and the variance σ in each of the state dimensions. (c) and (d) display the correlation matrices of the pick-up and place context at $t = 0.75$, respectively.

In Figure B.10a we observe the product of Gaussians produces the expected result for the x and y coordinates: the product's output is 'pulled' towards the frame with the lowest variance. Yet, the outcome of the z coordinate seems irregular; the product output around $t = 0.75$ lies below the GMR-output of the target frame.

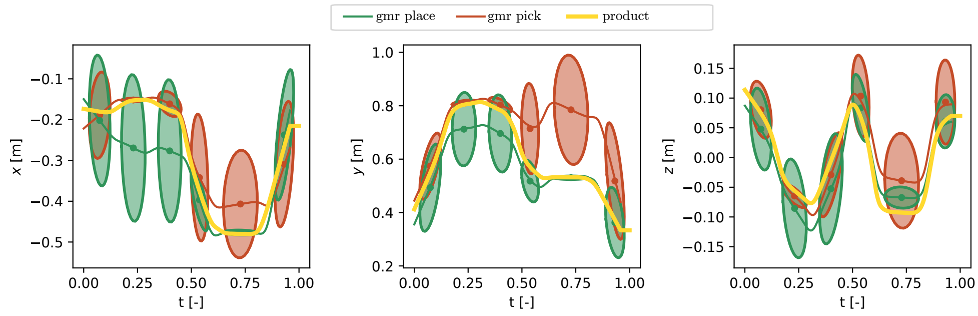
How can this behavior be explained? The Gaussian product trades-off both the GMR means thereby taking into account their covariance. Figure B.10b visualizes the outcome of this trade-off per dimension. If a context-dimension lies close to zero, the product was governed by this context (i.e. pick-up or place context). At the visualized time instance ($t = 0.75$), the product mostly governed by the place-frame. Only the ω_y and z dimensions show a relatively large error for the place context. Also note that the trade-off yields an equal error on ω_y for both pick-up and place frames.

The downward shift in the z dimension is mainly caused by the correlation and variance information of ω_y and z. The place-frame allows a relatively large variance for ω_z and shows a negative correlation with z. This combination permits a negative displacement in the z direction for a positive angle error ω_y. Although there is a similar correlation between the rotational axes of the pick-up frame and its z coordinate, their contributions is less severe because they have a high variance (and therefore lower cost in the trade-off).

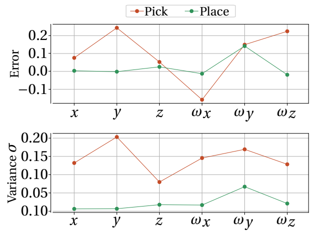
This behavior can damped by imposing a prior on the output of GMR. In Figure B.11a, we obtain the desired outcome by 'shrinking' the covariance matrix [167]. This is achieved by replacing the covariance outputed by GMR ($\boldsymbol{\Sigma}_{gmr}$) by

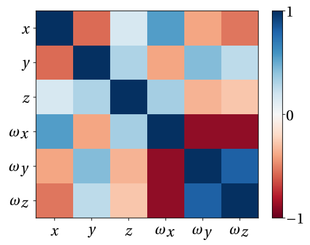
$$\hat{\boldsymbol{\Sigma}} = \lambda \boldsymbol{\Sigma}^d + (1-\lambda)\boldsymbol{\Sigma}_{gmr}, \qquad \text{(B.1)}$$

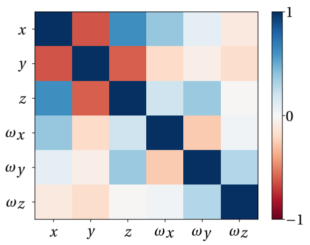
where $\boldsymbol{\Sigma}^d = \mathrm{diag}(\boldsymbol{\Sigma}_{gmr})$ only contains the diagonal components of $\boldsymbol{\Sigma}_{gmr}$. In effect, the shrinkage regularization lessens the correlation among the manifold dimensions. This is apparent in Figures B.11c and B.11d, where the correlation in the off-diagonal is slightly less compared to Figures B.10c and B.10d. As a result, the outcome of the product of Gaussians follows yields the expected behavior as illustrated in Figure B.11a.



(a) Spatial synthesis results

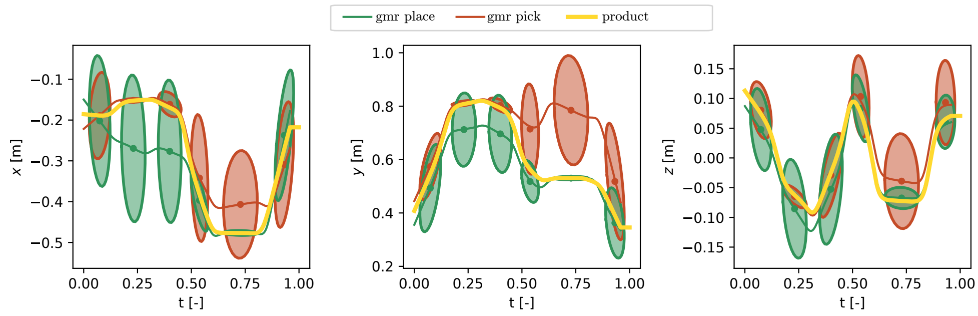




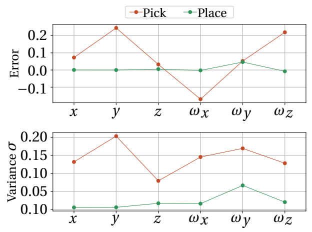


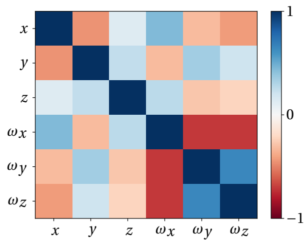
(b) Error (top) and variance (bottom) information (c) Correlation in pick-up context (d) Correlation in place context

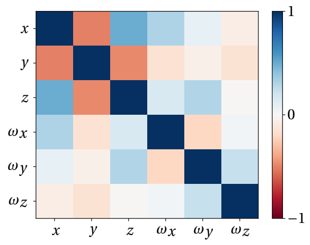
Figure B.10: Synthesis results without regularization. Refer to Section B.1.3 for a detailed description.



(a) Spatial synthesis results







(b) Error (top) and variance (bottom) information (c) Correlation in pick-up context (d) Correlation in place context

Figure B.11: Synthesis results with regularization ($\lambda = 0.2$). Refer to Section B.1.3 for a detailed description.

B.2. BOX TAPING

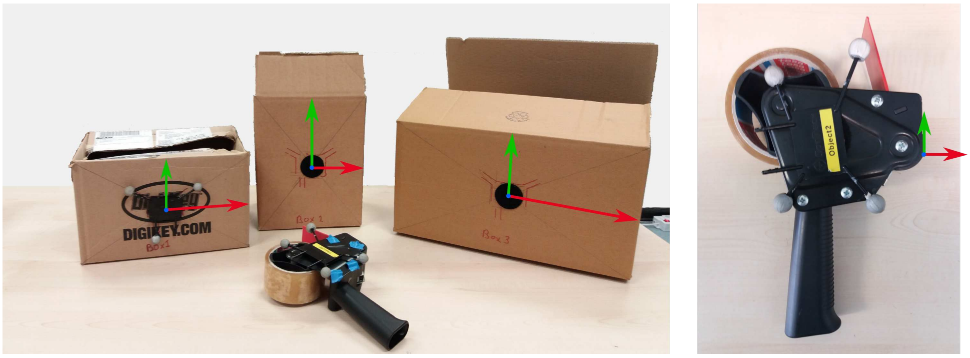


Figure B.12: Experimental setup used to record the box-tape data. Box sizes left-to-right: $0.175 \times 0.275 \times 0.215$, $0.260 \times 0.22 \times 0.16$, $0.22 \times 0.40 \times 0.31$ (H×W×D [m]). The colored lines and dot on each box indicate orientation and shape of the box frames. The axes x (Width/2), y (Height/2) and z(dot, Depth/2) are colored red, green and blue, respectively. The location of each frame lies inside the boxes at its geometrical center. The pose of the taping tool is measured at the taping contact point, and is visualized using the colored (orthonormal) reference frame.

Closing a carton box with a tape dispenser is a typical example of a generalizable task. The task involves recurrent features for boxes of different size and pose: the taping motion goes across the closing seam on top of the box, and is started and stopped about half-way the adjacent sides. This type of context is compatible with the affine context parameterization, as it can represent context with varying position, orientation and scaling. Furthermore, a proper orientation encoding is required in this application: the taping tool needs to be well-aligned with the seam, and rotate around the corners of the box. The Riemannian approach suits this purpose well, as no prior engineering is required on the orientation aspect. This opposed to using Euler angles, in which a suitable order of rotation needs to be defined; or axis-angle, where a suitable 'identity' is required to minimize potential distortions in angle measurements (see discussion in Section 2.2.2).

The objective of the box-tape application is to demonstrate the use of Riemannian TP-GMM with an affine context parameterization. Figure B.12 shows the components of the experimental setup. It consists of 3 boxes with different sizes. To simplify the experimental implementation, the setup is based on a marker based motion capture system. This system emulates object recognition to determine box size and orientation.

B.2.1. CONTEXT PARAMETERIZATION & MODELING

The task is demonstrated 3 times on boxes 2 & 3, and validated on boxes 1 & 2. The demonstration involves moving the tape tool to one adjacent side over the surface of the box, to the other adjacent side, while covering the closing seam. After this motion, the tool is returned to the initial position. In each demonstration, a temporal signal, and poses of the box and tape-tool are recorded. The demonstration data lie on the manifold $\mathcal{M}_{tape} = \mathbb{R} \times \mathbb{R}^3 \times \mathcal{S}^3 \times \mathbb{R}^3 \times \mathcal{S}^3$. Each demonstration is started and stopped by the demonstrator, and recorded using a sampling rate of 100 [Hz]. To improve temporal alignment of the demonstrations, excess data at the start and end of each demonstration is manually cut using a graphical interface, and the temporal signals are normalized.

The task context is defined by two frames: the box, and the pick-up location. The box frame is formalized using an affine transformation: $\boldsymbol{Q}_{box} \in \mathrm{GL}(3)$, $\boldsymbol{v}_{box} \in \mathbb{R}^3$. The pick-up location (pu) is defined using a rigid-body transformation: $\boldsymbol{R}_{pu} \in \mathrm{SO}(3)$, $\boldsymbol{v}_{pu} \in \mathbb{R}^3$. Together this yields the context parameterization

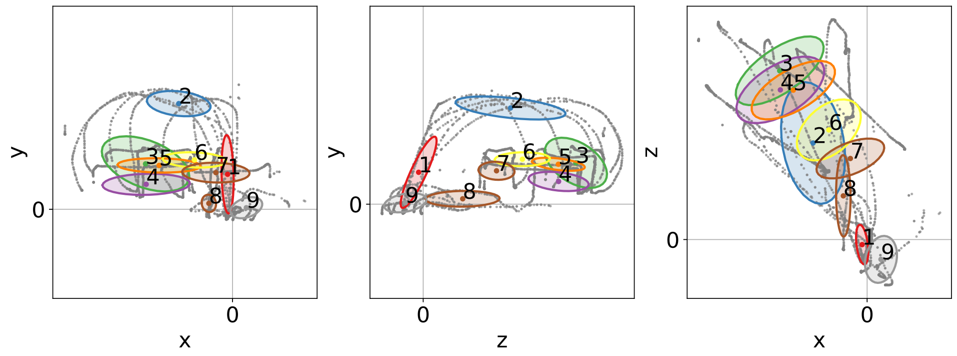
$$\boldsymbol{A} = \begin{bmatrix} t & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}_{pu} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I}_3 & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{Q}_{box} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I}_3 \end{bmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} 0 \\ \boldsymbol{v}_{pu} \\ \Psi_{\mathcal{S}^3}\left(\boldsymbol{R}_{pu}\right) \\ \boldsymbol{v}_{box} \\ \Psi_{\mathcal{S}^3}\left(\boldsymbol{Q}_{box}\right) \end{pmatrix}, \tag{B.2}$$

with $\Psi_{\mathcal{S}^3}(\cdot)$ as defined in Section 4.3.1, and $\boldsymbol{0}$ are matrices of appropriate size filled with zeros . The first element of $\boldsymbol{A}$ and $\boldsymbol{b}$ corresponds to the temporal signal. Through t, the duration of the reproduction can be modified. It can for example be set to the average duration of the demonstrations. In each demonstration, the pick-up location ($\boldsymbol{R}_{pu} \in \mathrm{SO}(3)$, $\boldsymbol{v}_{pu} \in \mathbb{R}^3$) is assumed fixed and defined as the first recorded tool pose.

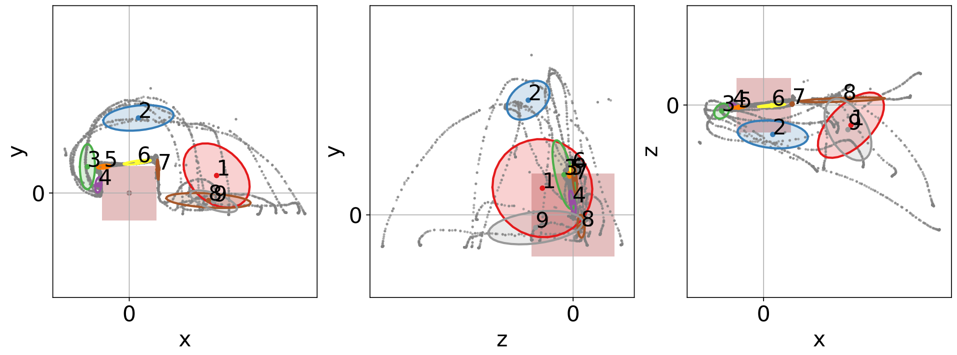
To encode the demonstration data, it is first projected in the individual frames. The projected data are then encoded in a Riemannian TP-GMM with 9 Gaussians (empirically set) on the manifold $\mathcal{M}_{tape}$. Encoding is performed using EM and initialized using K-means (see Appendix A.1). The demonstration data and model are visualized in Figure B.13. Note the affine context parameterization transformed the data in such a way that data acquired from different box shapes and poses overlap in the box frame (Figure B.13b). This is in contrast to the rigid-body transformation of the pick-up frame, which only aligns the data w.r.t. the pick-up pose, but does not perform any scaling.

B.2.2. SYNTHESIS

After modeling the taping motion, it can be synthesized in new context. Following Algorithm 4.2, the model is first contextualized based on the observed box pose and shape, and tape-tool pose. Then, GMR is performed using a temporal signal as input, to determine the desired states of the different frames. Per time step, the regression results are merged using the Gaussian product. Figure B.14 demonstrates two reproductions in new and previously encountered context. Note the relative horizontal and vertical stretch of the GMMs in Figures B.14b and B.14b. They show that GMM adapts to the box shape. Furthermore, note that the encoded orientation of the end-effector changes in a similar fashion for both box shapes.

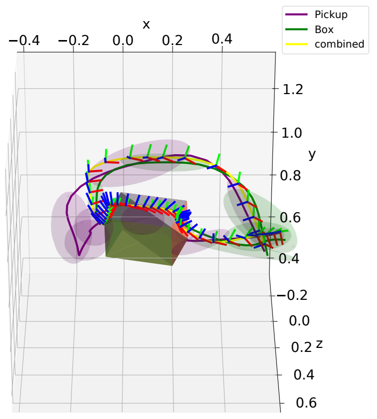


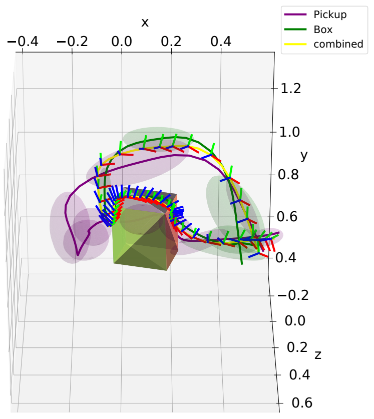
(a) Demonstration data and GMM visualized in the pick-up frame.



(b) Demonstration data and GMM visualized in the box frame.

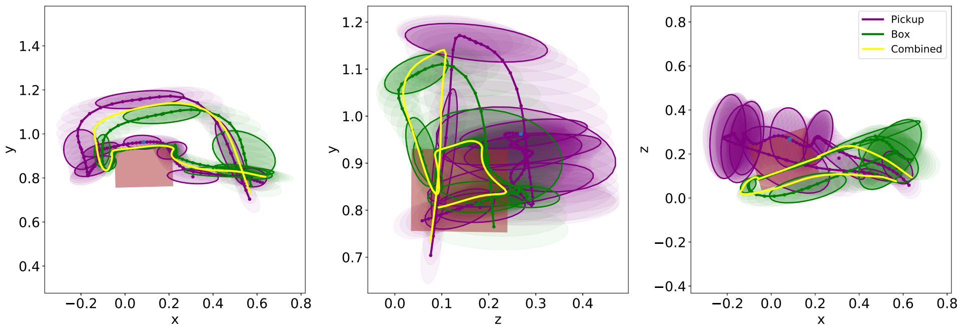
Figure B.13: Visualization of the data projected and TP-GMM in the pick-up frame (a) and box frame (b).





(a) Reproduction on box 1

(b) Reproduction on box2



(c) 2D projections of reproduction on box 1,

Figure B.14: Box taping reproductions in a previously encountered context (a, c), and new context (b). The 3D figures(a,b) show the position outcome of GMR for the box and pickup frame in green and purple, respectively. The outcome of the product of Gaussians (yellow), also demonstrates the resulting orientation using the 3 colored orthogonal lines. The cubic shape indicates the pose of the box. (c) shows 2D views of (a), where the box is depicted as a rectangle.

References

[1] S. Schaal, "Nonparametric regression for learning nonlinear transformations," in *Prerational Intelligence in Strategies, High-Level Processes and Collective Behavior*, H. Ritter and O. Holland, Eds. Dordrecht, The Netherlands: Kluwer Academic Press, 1999.

[2] A. G. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Secaucus, NJ, USA: Springer, 2008, pp. 1371–1394.

[3] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, 2009.

[4] A. G. Billard, S. Calinon, and R. Dillmann, "Learning from humans," in *Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Secaucus, NJ, USA: Springer, 2016, ch. 74, pp. 1995–2014, 2nd Edition.

[5] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2009.

[6] R. S. Sutton and A. G. Barto, *Reinforcement learning : an introduction*, ser. Adaptive computation and machine learning. Cambridge, MA, USA: MIT Press, 1998.

[7] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Philosophical Transaction of the Royal Society of London: Series B, Biological Sciences*, vol. 358, no. 1431, pp. 537–547, 2003.

[8] A. Ijspeert, J. Nakanishi, P. Pastor, H. Hoffmann, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Computation*, vol. 25, no. 2, pp. 328–373, 2013.

[9] D. M. Wolpert, J. Diedrichsen, and J. R. Flanagan, "Principles of sensorimotor learning," *Nature Reviews*, vol. 12, pp. 739–751, 2011.

[10] J. Kober, A. Wilhelm, E. Oztop, and J. Peters, "Reinforcement learning to adjust parametrized motor primitives to new situations," *Autonomous Robots*, vol. 33, no. 4, pp. 361–379, November 2012.

[11] S. Levine, N. Wagener, and P. Abbeel, "Learning contact-rich manipulation skills with guided policy search," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, May 2015, pp. 156–163.

[12] E. Todorov and M. I. Jordan, "Optimal feedback control as a theory of motor coordination," *Nature Neuroscience*, vol. 5, pp. 1226–1235, 2002.

[13] D. Sternad, M. E. Huber, and N. Kuznetsov, "Acquisition of novel and complex motor skills: stable solutions where intrinsic noise matters less," in *Progress in Motor Control.* Springer, 2014, pp. 101–124.

[14] N. Mansard and F. Chaumette, "Task sequencing for high-level sensor-based control," *IEEE Trans. on Robotics*, vol. 23, no. 1, pp. 60–72, 2007.

[15] E. Saltzman and J. Kelso, "Skilled actions: A task-dynamic approach." *Psychological review*, vol. 94, no. 1, p. 84, 1987.

[16] J. A. S. Kelso, "Synergies: Atoms of brain and behavior," in *Progress in Motor Control*, ser. Advances in Experimental Medicine and Biology, D. Sternad, Ed. Springer US, 2009, vol. 629, pp. 83–91.

[17] N. Hogan and D. Sternad, "Dynamic primitives of motor behavior," *Biological Cybernetics*, vol. 106, no. 11–12, pp. 727–739, 2012.

[18] S. Calinon, D. Bruno, and D. G. Caldwell, "A task-parameterized probabilistic model with minimal intervention control," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Hong Kong, China, May-June 2014, pp. 3339–3344.

[19] M. J. A. Zeestraten, S. Calinon, and D. G. Caldwell, "Variable duration movement encoding with minimal intervention control," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Stockholm, Sweden, May 2016, pp. 497–503.

[20] M. Mühlig, M. Gienger, S. Hellbach, J. J. Steil, and C. Goerick, "Task-level imitation learning using variance-based movement optimization," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA).* IEEE, 2009, pp. 1177–1184.

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

[22] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, "Probabilistic movement primitives," in *Advances in Neural Information Processing Systems (NIPS)*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 2616–2624.

[23] ——, "Using probabilistic movement primitives in robotics," *Autonomous Robots*, pp. 1–23, 2017.

[24] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Learning control policies for movement imitation and movement recognition," in *Neural Information Processing System (NIPS)*, vol. 15, 2003, pp. 1547–1554.

[25] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA).* IEEE, 2009, pp. 763–768.

[26] A. Ude, B. Nemec, T. Petrič, and J. Morimoto, "Orientation in cartesian space dynamic movement primitives," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Hong Kong, China, May-June 2014, pp. 2997–3004.

[27] A. Gams, B. Nemec, A. Ijspeert, and A. Ude, "Coupling movement primitives: Interaction with the environment and bimanual tasks," *IEEE Trans. on Robotics*, vol. 30, no. 4, pp. 816–830, Aug. 2014.

[28] R. F. Reinhart and J. J. Steil, "Efficient policy search in low-dimensional embedding spaces by generalizing motion primitives with a parameterized skill memory," *Autonomous Robots*, vol. 38, no. 4, pp. 331–348, 2015.

[29] A. Kramberger, A. Gams, B. Nemec, and A. Ude, "Generalization of orientational motion in unit quaternion space," in *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 808–813.

[30] B. Nemec, N. Likar, A. Gams, and A. Ude, "Bimanual human robot cooperation with adaptive stiffness control," in *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 607–613.

[31] A. Pervez and D. Lee, "Learning task-parameterized dynamic movement primitives using mixture of gmms," *Intelligent Service Robotics*, pp. 1–18, 2017.

[32] S. Calinon, F. Guenter, and A. G. Billard, "On learning the statistical representation of a task and generalizing it to various contexts," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Orlando, Florida, USA, May 2006, pp. 2978–2983.

[33] ——, "On learning, representing and generalizing a task in a humanoid robot," *IEEE Trans. on Systems, Man and Cybernetics, Part B*, vol. 37, no. 2, pp. 286–298, 2007.

[34] S. M. Khansari-Zadeh and A. Billard, "Imitation learning of globally stable nonlinear point-to-point robot motions using nonlinear programming," in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, 2010, pp. 2676–2683.

[35] ——, "Learning stable non-linear dynamical systems with Gaussian mixture models," *IEEE Trans. on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[36] J. R. Medina, D. Lee, and S. Hirche, "Risk-sensitive optimal feedback control for haptic assistance," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, May 2012, pp. 1025–1031.

[37] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. Cambridge, MA, USA: MIT Press, 2006.

[38] M. Lang and S. Hirche, "Computationally efficient rigid-body gaussian process for motion dynamics," *IEEE Robotics and Automation Letters (RA-L)*, vol. 2, no. 3, pp. 1601–1608, 2017.

[39] J. Umlauft, Y. Fanger, and S. Hirche, "Bayesian uncertainty modeling for programming by demonstration," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 6428–6434.

[40] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, “Statistical dynamical systems for skills acquisition in humanoids,” in *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*, Osaka, Japan, 2012, pp. 323–329.

[41] J. Silvério, L. Rozo, S. Calinon, and D. G. Caldwell, “Learning bimanual end-effector poses from demonstrations using task-parameterized dynamical systems,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, Sept.-Oct. 2015, pp. 464–470.

[42] S. Kim, R. Haschke, and H. Ritter, “Gaussian mixture model for 3-DoF orientations,” *Robotics and Autonomous Systems*, vol. 87, pp. 28–37, 2017.

[43] E. Rueckert, J. Mundo, A. Paraschos, J. Peters, and G. Neumann, “Extracting low-dimensional control variables for movement primitives,” in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Seattle, WA, USA, 2015, pp. 1511–1518.

[44] S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain, “Gaussian processes for time-series modelling,” *Philosophical Trans. of the Royal Society A*, vol. 371, no. 1984, pp. 1–25, 2012.

[45] S. Hauberg, O. Freifeld, and M. J. Black, “A geometric take on metric learning,” in *Advances in Neural Information Processing Systems (NIPS)*, 2012, pp. 2024–2032.

[46] B. Kulis, “Metric learning: A survey,” *Foundations and Trends in Machine Learning*, vol. 5, no. 4, pp. 287–364, 2013.

[47] X. Pennec, “Intrinsic statistics on Riemannian manifolds: Basic tools for geometric measurements,” *Journal of Mathematical Imaging and Vision*, vol. 25, no. 1, pp. 127–154, 2006.

[48] M. J. A. Zeestraten, I. Havoutis, J. Silvério, S. Calinon, and D. G. Caldwell, “An approach for imitation learning on Riemannian manifolds,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 2, no. 3, pp. 1240–1247, June 2017.

[49] L. Rozo, N. Jaquier, S. Calinon, and D. G. Caldwell, “Learning manipulability ellipsoids for task compatibility in robot manipulation,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Vancouver, Canada, September 2017.

[50] N. Jaquier and S. Calinon, “Gaussian mixture regression on symmetric positive definite matrices manifolds: Application to wrist motion estimation with sEMG,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Vancouver, Canada, September 2017.

[51] E. Simo-Serra, C. Torras, and F. Moreno-Noguer, “3D human pose tracking priors using geodesic mixture models,” *International Journal of Computer Vision*, vol. 122, no. 2, pp. 388–408, 2017.

[52] G. Dubbelman, “Intrinsic statistical techniques for robust pose estimation,” Ph.D. dissertation, University of Amsterdam, September 2011.

[53] K. C. Wolfe and M. Mashner, “Bayesian fusion on lie groups,” *Journal of Algebraic Statistics*, vol. 2, no. 1, pp. 75–97, 2011.

[54] B. B. Ready, “Filtering techniques for pose estimation with applications to unmanned air vehicles,” Ph.D. dissertation, Brigham Young University-Provo, November 2012.

[55] S. Hauberg, F. Lauze, and K. S. Pedersen, “Unscented Kalman filtering on Riemannian manifolds,” *Journal of mathematical imaging and vision*, vol. 46, no. 1, pp. 103–120, 2013.

[56] T. D. Barfoot and P. T. Furgale, “Associating uncertainty with three-dimensional poses for use in estimation problems,” *IEEE Trans. on Robotics*, vol. 30, no. 3, pp. 679–693, June 2014.

[57] “SMART-E International Training Network (ITN),” www.smart-e-mariecurie.eu, accessed: 10 October 2017.

[58] “European project: AutoMAP,” http://www.euroc-project.eu/index.php?id=automap, accessed: 10 October 2017.

[59] M. Lang, M. Kleinsteuber, O. Dunkley, and S. Hirche, “Gaussian process dynamical models over dual quaternions,” in *European Control Conference (ECC)*, 2015, pp. 2847–2852.

[60] W. Feiten, M. Lang, and S. Hirche, “Rigid Motion Estimation using Mixtures of Projected Gaussians,” in *16th Intl. Conf. on Information Fusion (FUSION)*, 2013, pp. 1465–1472.

[61] S. Chiaverini, “Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators,” *IEEE Trans. on Robotics and Automation*, vol. 13, no. 3, pp. 398–410, 1997.

[62] B. Siciliano and J. J. E. Slotine, “A general framework for managing multiple tasks in highly redundant robotic systems,” in *Advanced Robotics, 1991. 'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, June 1991, pp. 1211–1216 vol.2.

[63] J.-O. Kim, M. Wayne, and P. K. Khosla, “Exploiting redundancy to reduce impact force,” *Journal of Intelligent and Robotic Systems*, vol. 9, no. 3, pp. 273–290, 1994.

[64] A. De Luca and L. Ferrajoli, “Exploiting robot redundancy in collision detection and reaction,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Nice, France, September 2008.

[65] J. M. Lee, *Riemannian Manifolds - An Introduction to Curvature*, 1st ed. New York: Springer-Verlag, 1997.

[66] J. Jost, *Riemannian geometry and geometric analysis*, ser. Universitext. Berlin: Springer, 2002.

[67] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds.* Princeton, NJ: Princeton University Press, 2008.

[68] R. M. Murray, S. S. Sastry, and Z. Li, *A Mathematical Introduction to Robotic Manipulation.* Boca Raton, FL, USA: CRC Press, Inc., 1994.

[69] A. L. Schwab and J. Meijaard, "How to draw euler angles and utilize euler parameters," in *Proc. the IDETC/CIE*, Sep. 2006, pp. 259–265.

[70] K. B. Petersen and M. S. Pedersen, "The matrix cookbook," Nov 2012.

[71] A. L. Schwab, "Quaternions, finite rotation and euler parameters," *Cornell University Notes, Ithaca NY*, 2002.

[72] K. V. Mardia and P. E. Jupp, *Directional Statistics.* Wiley, 1999.

[73] M. Malekzadeh, J. Queißer, and J. J. Steil, "Imitation learning for a continuum trunk robot," in *Proc. of the European Symp. on Artificial Neural Networks, Computational Intelligence and Machine Learning*, M. Verleysen, Ed., 2017.

[74] E. Gribovskaya and A. Billard, "Learning nonlinear multi-variate motion dynamics for real-time position and orientation control of robotic manipulators," in *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids).* IEEE, 2009, pp. 472–477.

[75] S. Kim, A. Shukla, and A. Billard, "Catching objects in flight," *IEEE Trans. on Robotics*, vol. 30, no. 5, pp. 1049–1065, Oct. 2014.

[76] J. Glover and L. P. Kaelbling, "Tracking 3-d rotations with the quaternion bingham filter," MIT, Tech. Rep., March 2013.

[77] J. M. Glover, "The quaternion bingham distribution, 3d object detection, and dynamic manipulation," Ph.D. dissertation, Massachusetts Institute of Technology, 2014.

[78] G. Kurz, I. Gilitschenski, S. Julier, and U. D. Hanebeck, "Recursive bingham filter for directional estimation involving 180 degree symmetry," *Journal of Advances in Information Fusion*, vol. 9, no. 2, pp. 90–105, 2014.

[79] I. Gilitschenski, G. Kurz, S. J. Julier, and U. D. Hanebeck, "Unscented orientation estimation based on the bingham distribution," *IEEE Trans. on Automatic Control*, vol. 61, no. 1, pp. 172–177, 2016.

[80] S. F. Su and C. S. G. Lee, "Uncertainty manipulation and propagation and verification of applicability of actions in assembly tasks," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA).* IEEE, 1991, pp. 2471–2476.

[81] Y. Wang and G. S. Chirikjian, "Error propagation on the euclidean group with applications to manipulator kinematics," *IEEE Trans. on Robotics*, vol. 22, no. 4, pp. 591–602, 2006.

[82] C. Hertzberg, R. Wagner, U. Frese, and L. Schröder, “Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds,” *Information Fusion*, vol. 14, no. 1, pp. 57–77, 2013.

[83] G. Bourmaud, R. Mégret, M. Arnaudon, and A. Giremus, “Continuous-discrete extended kalman filter on matrix lie groups using concentrated gaussian distributions,” *Journal of Mathematical Imaging and Vision*, vol. 51, no. 1, pp. 209–228, 2015.

[84] M. Zefran, V. Kumar, and C. Croke, “Metrics and connections for rigid-body kinematics,” *International Journal of Robotics Research*, vol. 18, no. 2, pp. 243–258, 1999.

[85] R. Featherstone, *Rigid Body Dynamics Algorithms*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2008.

[86] F. Park, J. Bobrow, and S. Ploen, “A lie group formulation of robot dynamics,” *The International Journal of Robotics Research*, vol. 14, no. 6, pp. 609–618, 1995.

[87] F. Park, “Distance metrics on the rigid-body motions with applications to mechanism design,” *ASME Journal of Mechanism Design*, vol. 117, no. 1, pp. 48–54, Jan. 1995.

[88] J. Loncaric, “Geometrical analysis of compliant mechanisms in robotics (euclidean group, elastic systems, generalized springs,” Ph.D. dissertation, Harvard University, Cambridge, MA, USA, 1985.

[89] F. C. Park and R. W. Brockett, “Kinematic dexterity of robotic mechanisms,” *The International Journal of Robotics Research*, vol. 13, no. 1, pp. 1–15, 1994.

[90] H. Karcher, “Riemannian center of mass and mollifier smoothing,” *Communications on Pure and Applied Mathematics*, vol. 30, no. 5, pp. 509–541, 1977.

[91] W. S. Kendall, “Probability, convexity, and harmonic maps with small image i: Uniqueness and fine existence,” *Proceedings of the London Mathematical Society*, vol. s3-61, no. 2, pp. 371–406, 1990.

[92] H. Karcher, “Riemannian center of mass and so called karcher mean,” *arXiv preprint arXiv:1407.2087*, 2014.

[93] O. Freifeld, S. Hauberg, and M. J. Black, “Model transport: Towards scalable transfer learning on manifolds,” in *Proc. CVPR*, Columbus, OH, USA, June 2014, pp. 1378–1385.

[94] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer, 2006.

[95] M. J. A. Zeestraten, I. Havoutis, S. Calinon, and D. G. Caldwell, “Learning task-space synergies controllers using riemannian manifolds,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Vancouver, Canada, September 2017.

[96] M. L. Latash, J. P. Scholz, and G. Schöner, “Toward a new theory of motor synergies,” *Motor control*, vol. 11, no. 3, pp. 276–308, 2007.

[97] A. Saccon, J. Hauser, and A. P. Aguiar, “Optimal control on lie groups: The projection operator approach,” *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2230–2245, Sept 2013.

[98] M. M. Marinho, L. F. C. Figueredo, and B. V. Adorno, “A dual quaternion linear-quadratic optimal controller for trajectory tracking,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Sept 2015, pp. 4047–4052.

[99] X. Wang and C. Yu, “Unit dual quaternion-based feedback linearization tracking problem for attitude and position dynamics,” *Systems & Control Letters*, vol. 62, no. 3, pp. 225–233, 2013.

[100] L. Rozo, S. Calinon, D. G. Caldwell, P. Jimenez, and C. Torras, “Learning collaborative impedance-based robot behaviors,” in *Proc. AAAI Conference on Artificial Intelligence*, Bellevue, Washington, USA, 2013, pp. 1422–1428.

[101] M. Saveriano and D. Lee, “Learning motion and impedance behaviors from human demonstrations,” in *Intl. Conf. on Ubiquitous Robots and Ambient Intelligence (URAI)*, Nov 2014, pp. 368–373.

[102] K. Kronander and A. Billard, “Learning compliant manipulation through kinesthetic and tactile human-robot interaction,” *IEEE Transactions on Haptics*, vol. 7, no. 3, pp. 367–380, 2014.

[103] A. E. Bryson, *Dynamic optimization.* Addison Wesley Longman Menlo Park, 1999.

[104] S. P. Bhat and D. S. Bernstein, “A topological obstruction to continuous global stabilization of rotational motion and the unwinding phenomenon,” *Systems & Control Letters*, vol. 39, no. 1, pp. 63–70, 2000.

[105] N. A. Chaturvedi, A. K. Sanyal, and N. H. McClamroch, “Rigid-body attitude control,” *IEEE Control Systems Magazine*, vol. 31, no. 3, pp. 30–51, June 2011.

[106] F. Bullo and R. Murray, “Proportional derivative (PD) control on the euclidean group,” in *European Control Conference*, vol. 2, 1995, pp. 1091–1097.

[107] A. Ude, A. Gams, T. Asfour, and J. Morimoto, “Task-specific generalization of discrete and periodic dynamic movement primitives,” *IEEE Trans. on Robotics*, vol. 26, no. 5, pp. 800–815, 2010.

[108] G. J. Maeda, G. Neumann, M. Ewerton, R. Lioutikov, O. Kroemer, and J. Peters, “Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks,” *Autonomous Robots*, vol. 41, no. 3, pp. 593–612, Mar 2017.

[109] S. Brandi, O. Kroemer, and J. Peters, “Generalizing pouring actions between objects using warped parameters,” in *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*. IEEE, 2014, pp. 616–621.

[110] A. Alissandrakis, C. L. Nehaniv, and K. Dautenhahn, “Correspondence mapping induced state and action metrics for robotic imitation,” *IEEE Trans. on Systems, Man and Cybernetics, Part B*, vol. 37, no. 2, pp. 299–307, 2007.

[111] C. Bowen, G. Ye, and R. Alterovitz, “Asymptotically optimal motion planning for learned tasks using time-dependent cost maps,” *Automation Science and Engineering, IEEE Transactions on*, vol. 12, no. 1, pp. 171–182, 2015.

[112] Y. Yang, V. Ivan, and S. Vijayakumar, “Real-time motion adaptation using relative distance space representation,” in *Proc. Intl Conf. on Advanced Robotics (ICAR)*, July 2015, pp. 21–27.

[113] J. De Schutter, “Invariant description of rigid body motion trajectories,” *Journal of Mechanisms and Robotics*, vol. 2, no. 1, pp. 1–9, 2010.

[114] M. Vochten, T. De Laet, and J. De Schutter, “Generalizing demonstrated motions and adaptive motion generation using an invariant rigid body trajectory representation,” in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 234–241.

[115] D. Lee, R. Soloperto, and M. Saveriano, “Bidirectional invariant representation of rigid body motions and its application to gesture recognition and reproduction,” *Autonomous Robots*, pp. 1–21, 2017.

[116] B. Nemec, M. Tamošiūnaitė, F. Wörgötter, and A. Ude, “Task adaptation through exploration and action sequencing,” in *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*. IEEE, 2009, pp. 610–616.

[117] B. C. da Silva, G. Konidaris, and A. G. Barto, “Learning parameterized skills,” in *Proc. Intl Conf. on Machine Learning (ICML)*, 2012.

[118] S. Levine, C. Finn, T. Darrell, and P. Abbeel, “End-to-end training of deep visuomotor policies,” *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016.

[119] W. Montgomery, A. Ajay, C. Finn, P. Abbeel, and S. Levine, “Reset-free guided policy search: efficient deep reinforcement learning with stochastic initial states,” in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3373–3380.

[120] S. M. Khansari-Zadeh and A. Billard, “Learning control Lyapunov function to ensure stability of dynamical system-based robot reaching motions,” *Robotics and Autonomous Systems*, vol. 62, no. 6, pp. 752–765, 2014.

[121] A. D. Wilson and A. F. Bobick, “Parametric hidden Markov models for gesture recognition,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 21, no. 9, pp. 884–900, 1999.

[122] A. P. Shon, D. Verma, and R. Rao, “Active imitation learning,” in *Proc. AAAI Conference on Artificial Intelligence*, JAN 2007, pp. 756–762.

[123] G. Maede, M. Ewerton, T. Osa, and J. Peters, "Active imitation learning," in *Proc. IFRR/JMLR Annual Conf. On Robot Learning (CORL)*, Nov 2017.

[124] J. Hollerbach, "Dynamic scaling of manipulator trajectories," *ASME Journal of Dynamic Systems, Measurement and Control*, vol. 106, no. 1, pp. 102–106, March 1984.

[125] G. Schwarz, "Estimating the dimension of a model," *Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[126] S. P. Chatzis, D. Korkinof, and Y. Demiris, "A nonparametric Bayesian approach toward robot learning by demonstration," *Robotics and Autonomous Systems*, vol. 60, no. 6, pp. 789–802, 2012.

[127] I. Havoutis and S. Calinon, "Supervisory teleoperation with online learning and optimal control," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Singapore, May-June 2017, pp. 1534–1540.

[128] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77:2, pp. 257–285, February 1989.

[129] C. Passenberg, A. Peer, and M. Buss, "A survey of environment-, operator-, and task-adapted controllers for teleoperation systems," *Mechatronics*, vol. 20, no. 7, pp. 787–801, October 2010.

[130] H. Boessenkool, D. A. Abbink, C. J. M. Heemskerk, F. C. T. van der Helm, and J. G. W. Wildenbeest, "A task-specific analysis of the benefit of haptic shared control during telemanipulation," *IEEE Transactions on Haptics*, vol. 6, no. 1, pp. 2–12, June 2013.

[131] C. J. Pérez-del Pulgar, J. Smisek, V. F. Muñoz, and A. Schiele, "Using learning from demonstration to generate real-time guidance for haptic shared control," in *Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on*. IEEE, 2016, pp. 3205–3210.

[132] L. B. Rosenberg, "Virtual fixtures: Perceptual tools for telerobotic manipulation," in *Proc. IEEE Intl Symp. Virtual Reality (VRAIS)*, Sep 1993, pp. 76–82.

[133] J. J. Abbott, P. Marayong, and A. M. Okamura, "Haptic virtual fixtures for robot-assisted manipulation," in *Proc. Intl Symp. on Robotics Research (ISRR)*. Springer Berlin Heidelberg, 2007, pp. 49–64.

[134] J. van Oosterhout, J. G. W. Wildenbeest, H. Boessenkool, C. J. M. Heemskerk, M. R. de Baar, F. C. T. van der Helm, and D. A. Abbink, "Haptic shared control in telemanipulation: Effects of inaccuracies in guidance on task execution," *IEEE transactions on haptics*, vol. 8, no. 2, pp. 164–175, 2015.

[135] D. A. Abbink, M. Mulder, and E. R. Boer, "Haptic shared control: smoothly shifting control authority?" *Cognition, Technology & Work*, vol. 14, no. 1, pp. 19–28, 2012.

[136] S. Park, R. D. Howe, and D. F. Torchiana, “Virtual fixtures for robotic cardiac surgery,” in *Proc. Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, W. J. Niessen and M. A. Viergever, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 1419–1420.

[137] G. Raiola, X. Lamy, and F. Stulp, “Co-manipulation with multiple probabilistic virtual guides,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, Sep. 2015, pp. 7–13.

[138] D. Katzourakis, M. Alirezaei, J. C. de Winter, M. Corno, R. Happee, A. Ghaffari, and R. Kazemi, “Shared control for road departure prevention,” in *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on.* IEEE, 2011, pp. 1037–1043.

[139] H. Saeidi, F. McLane, B. Sadrfaidpour, E. Sand, S. Fu, J. Rodriguez, J. Wagner, and Y. Wang, “Trust-based mixed-initiative teleoperation of mobile robots,” in *American Control Conference (ACC), 2016.* IEEE, 2016, pp. 6177–6182.

[140] C. Ezeh, P. Trautman, L. Devigne, V. Bureau, M. Babel, and T. Carlson, “Probabilistic vs linear blending approaches to shared control for wheelchair driving,” in *IEEE Int. Conf. on Rehabilitation Robotics, ICORR’17*, 2017.

[141] S. Bodenstedt, N. Padoy, and G. D. Hager, “Learned partial automation for shared control in tele-robotic manipulation.” in *AAAI Fall Symposium: Robots Learning Interactively from Human Teachers*, 2012.

[142] I. Havoutis and S. Calinon, “Learning assistive teleoperation behaviors from demonstration,” in *Proc. IEEE Intl Symp. on Safety, Security and Rescue Robotics*, Lausanne, Switzerland, October 2016, pp. 258–263.

[143] F. Abi-Farraj, T. Osa, N. Pedemonte, J. Peters, G. Neumann, and P. G. Robuffo, “A learning-based shared control architecture for interactive task execution,” in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2017.

[144] A. K. Tanwani and S. Calinon, “Learning robot manipulation tasks with task-parameterized semi-tied hidden semi-Markov model,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 1, no. 1, pp. 235–242, January 2016.

[145] R. Assmann, M. Magistris, O. Aberle, M. Mayer, F. Ruggiero, J. Jiménez, S. Calatroni, A. Ferrari, G. Bellodi, I. Kurochkin *et al.*, “The final collimation system for the lhc,” CERN, Tech. Rep., 2006.

[146] F. Bullo and A. D. Lewis, *Geometric Control of Mechanical Systems.* Springer-Verlag, 2004, vol. 49.

[147] N. Ratliff, M. Toussaint, and S. Schaal, “Understanding the geometry of workspace obstacles in motion optimization,” in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2015, pp. 4202–4209.

[148] P. D. Neilson, M. D. Neilson, and R. T. Bye, “A riemannian geometry theory of human movement: The geodesic synergy hypothesis,” *Human movement science*, vol. 44, pp. 42–72, 2015.

[149] T. Flash and A. A. Handzel, “Affine differential geometry analysis of human arm movements,” *Biological cybernetics*, vol. 96, no. 6, pp. 577–601, 2007.

[150] F. C. Park, “Some fundamental problems in robotics: a geometric perspective,” 2015.

[151] S.-Z. Yu and H. Kobayashi, “Practical implementation of an efficient forward-backward algorithm for an explicit-duration hidden Markov model,” *IEEE Trans. on Signal Processing*, vol. 54, no. 5, pp. 1947–1951, 2006.

[152] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for linear and hybrid systems.* Cambridge University Press, 2015, in preparation.

[153] V. Gómez, H. J. Kappen, J. Peters, and G. Neumann, “Policy search for path integral control,” in *Proc. European Conf. on Machine Learning and Knowledge Discovery in Databases*, vol. 8724. New York, NY, USA: Springer-Verlag New York, Inc., 2014, pp. 482–497.

[154] E. Theodorou, J. Buchli, and S. Schaal, “A generalized path integral control approach to reinforcement learning,” *J. Mach. Learn. Res.*, vol. 11, pp. 3137–3181, December 2010.

[155] E. Todorov, “Efficient computation of optimal actions,” *Proc. of the National Academy of Sciences of the United States of America*, vol. 106, no. 28, pp. 11 478–11 483, 2009.

[156] N. Aghasadeghi and T. Bretl, “Maximum entropy inverse reinforcement learning in continuous state spaces with path integrals,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, 2011, pp. 1561–1566.

[157] S. Levine and V. Koltun, “Continuous inverse optimal control with locally optimal examples,” in *Intl Conf. on Machine Learning (ICML)*, 2012.

[158] M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal, “Learning objective functions for manipulation,” in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2013, pp. 1331–1336.

[159] A. Doerr, N. Ratliff, J. Bohg, M. Toussaint, and S. Schaal, “Direct loss minimization inverse optimal control,” in *Proc. Robotics: Science and Systems (R:SS)*, Rome, Italy, July 2015, pp. 1–9.

[160] P. Englert and M. Toussaint, “Inverse KKT–learning cost functions of manipulation tasks from demonstrations,” in *Proc. Intl Symp. on Robotics Research (ISRR)*, 2015.

[161] C. Finn, S. Levine, and P. Abbeel, “Guided cost learning: Deep inverse optimal control via policy optimization,” in *Proceedings of 33rd International Conference on Machine Learning (ICML)*, June 2016, pp. 49–58.

[162] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the Royal Statistical Society B*, vol. 39, no. 1, pp. 1–38, 1977.

[163] E. Icer and M. Althoff, “Cost-optimal composition synthesis for modular robots,” in *Proc. of the IEEE Multi-Conference on Control Applications (CCA)*, September 2016, pp. 1408–1413.

[164] E. Icer, A. Giusti, and M. Althoff, “A task-driven algorithm for configuration synthesis of modular robots,” in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, May 2016, pp. 5203–5209.

[165] A. Giusti and M. Althoff, “Automatic centralized controller design for modular and reconfigurable robot manipulators,” in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, September 2015, pp. 3268–3275.

[166] ——, “On-the-fly control design of modular robot manipulators,” *IEEE Trans. on Control Systems Technology*, vol. –, no. –, pp. –, – 2017 (in press).

[167] K. P. Murphy, *Machine learning: a probabilistic perspective.* Cambridge, MA, USA: MIT Press, 2012.

Curriculum Vitæ

M.J.A.(Martijn) Zeestraten

02-03-1987	Born in De Lier, the Netherlands.

Education

2004–2009	Bachelor in Mechanical Engineering Technische Hogeschool Rijswijk
2010	Pre-Master Technische Universiteit Delft
2011–2013	Master in Mechanical Engineering (with distinction) Technische Universiteit Delft

Student Supervision

Winter 2016	Nikol Guljelmović Task Parameter Inference in Human-Robot Interaction Master Thesis, TU Delft
Winter 2017 (on going)	Stijn Seuren Recognizing and correcting control disagreement in shared control Master Thesis, TU Delft

Review Activities

The International Joint Conference on Artificial Intelligence (IJCAI), 2017
IEEE International Conference on Multisensor Fusion and Integration of Intelligent Systems (MFI), 2017
IEEE RAS International Conference on Humanoid Robots (HUMANOIDS), 2017
IEEE International Conference on Robotics and Automation (ICRA), 2017–2018
IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017
IEEE Transactions on Haptics, 2018

List of Publications

Journals papers (accepted)

- **M. J. A. Zeestraten**, I. Havoutis, J. Silvério, S. Calinon, and D. G. Caldwell, *An approach for imitation learning on Riemannian manifolds*, *IEEE Robotics and Automation Letters (RA-L)*, vol. 2, no. 3, pp. 1240–1247, June 2017.

Journals papers (submitted)

- A. Giusti, **M. J. A. Zeestraten**, E. Içer, A. Pereira, S. Calinon, D. G. Caldwell, and M. Althoff *Towards Flexible Automation Driven by Demonstration* (submitted).

- **M. J. A. Zeestraten**, I. Havoutis, S. Calinon, and D. G. Caldwell, *Programming by Demonstration for Shared Control with an Application in Teleoperation* (submitted).

Conference papers (peer reviewed, published)

- **M. J. A. Zeestraten**, I. Havoutis, S. Calinon, and D. G. Caldwell, *Learning Task-Space Synergy Controllers using Riemannian Manifolds*, in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Vancouver, Canada, September 2017.

- **M. J. A. Zeestraten**, A. Pereira, A. Althoff, and S. Calinon, *Online Motion Synthesis with Minimal Intervention Control and Formal Safety Guarantees* Proc. IEEE Intl Conf. on Systems, Man, and Cybernetics (SMC) , 2116–2121 (2016).

- **M. J. A. Zeestraten**, S. Calinon, and D. G. Caldwell, *Variable duration movement encoding with minimal intervention control*, in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Stockholm, Sweden, May 2016 pp. 497–503.