Programming by Demonstration for Shared Control with an Application in Teleoperation

Martijn J.A. Zeestraten[1], Ioannis Havoutis[2], Sylvain Calinon[3,1]

***Abstract*—Shared control strategies can improve task-performance in teleoperation. In such systems automation guides or corrects a human operator. The amount of correction or guidance that is provided is denoted the *level of automation*. As the variety of teleoperation tasks is large, manually specifying the underlying automation is time consuming. In this work, we present an approach to program this automated system by demonstration. Our approach determines the level of automation online, by combining the confidence of automation and teleoperator. We present particular implementations of our approach for haptic shared control and state shared control. The method is evaluated in a user study. Although the subjects indicated they preferred the learned shared control strategies, teleoperation performance did not improve our metric (task execution time).**

***Index Terms*—Learning and Adaptive Systems, Probability and Statistical Methods, Teleoperation, Shared Control**

I. 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 a *feeling of presence* [1].

Virtual assistance can further improve the teleoperator performance [2], [3]. 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 [4], [5]; its use significantly improves task performance [4], [6], and reduces mental workload [5].

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 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 [5]. 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 Programming by Demonstration (PbD) [7] can alleviate this problem, and propose to program virtual fixtures by demonstration.

Depending on the implementation, virtual fixtures can either be seen as a form of *semi-autonomous control* or *shared control*. In this paper, 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* state variables are *jointly* controlled by the human and assistive system. The weighting of the control inputs determines the level of automation.

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) [5], [8], where the intentions are mixed at the operator interface through haptic interaction, see for example [3], [4], [9], [10]. 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-

Manuscript received: September, 8, 2017; Revised December, 13, 2017; Accepted January, 22, 2018.

This paper was recommended for publication by Dongheui Lee upon evaluation of the Associate Editor and Reviewers' comments. The research leading to these results has received funding from the People Programme (Marie Curie Actions) of the European Union's 7th Framework Programme FP7/2007-2013/ under REA grant agreement number 608022. It is also supported by the DexROV Project through the EC Horizon 2020 programme (Grant #635491). I. Havoutis was supported by the UK EPSRC grants EP/M019918/1, EP/R026084/1 and EP/R026173/1.

[1] Department of Advanced Robotics, Istituto Italiano di Tecnologia, Via Morego 30, 16163 Genova, Italy. martijn.zeestraten@iit.it

[2] Oxford Robotics Institute, Department of Engineering Science, University of Oxford, United Kingdom. ioannis@robots.ox.ac.uk

[3] Idiap Research Institute, Rue Marconi 19, 1920 Martigny, Switzerland. sylvain.calinon@idiap.ch

Digital Object Identifier (DOI): see top of this page.

mixing shared control [11], mixed initiative control [12], input blending control [13] or policy blending [14]. 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.* [10] 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.* [15] 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.* [16] 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.* [17] 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.* [3] 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).

Pervez *et al.* [18] highlight alignment difficulties faced by PbD when handling multiple or partial demonstrations. These arise due to the presence of a temporal signal. In the approach that we present, we avoid the encoding of temporal information. We find temporally driven motions not well-suited for assistive teleoperation, as they force the teleoperator to follow a pre-defined temporal behavior.

Our work is similar to Havoutis *et al.* [16]. We advance it in two directions. First, our proposed approach relies on the Riemannian framework presented in [19], [20]. 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.* [16] only considered SSC.

This paper is organized as follows: First, our Riemannian approach for PbD [19] is reviewed in Section II. Section III introduces the shared control strategies HSC and SSC in more detail, and discusses how they are programmed by demonstration. Finally, we evaluate and discuss the approach in sections IV and V.

II. PRELIMINARIES

The methods that are presented in this paper involve control and statistical encoding of end-effector poses: elements of a Riemannian manifold. Riemannian manifolds are not generally Euclidean, and common approaches for control and statistics cannot be directly applied to Riemannian data. In previous work, we presented a probabilistic framework for PbD [19] and a Linear Quadratic Regulator (LQR) [20] for Riemannian data. This section reviews the concepts of statistics and LQR on Riemannian manifolds.

We start with the geometric concepts of distance and straightness. The distance between two points in a Euclidean space corresponds to the length of the straight line (segment) connecting them. A *geodesic* is the generalization of the straight line from a Euclidean space to a Riemannian manifold. Similarly to Euclidean spaces, the distance between two points on a Riemannian manifold corresponds to the length of the *geodesic* (segment) connecting them.

At each point $\boldsymbol{p}$ of the d-dimensional Riemannian Manifold one can define a Euclidean tangent space $\mathbb{R}^d$. We indicate elements of the manifold in bold and elements on the tangent space in fraktur typeface, i.e. $\boldsymbol{p} \in \mathcal{M}$ and $\mathfrak{g} \in \mathcal{T}_{\boldsymbol{p}}\mathcal{M}$. The *exponential map* $\mathrm{Exp} : \mathcal{T}_{\boldsymbol{p}}\mathcal{M} \to \mathcal{M}$ defines a local mapping between the tangent space and manifold. $\mathrm{Exp}_{\boldsymbol{p}}(\mathfrak{g})$ maps $\mathfrak{g} \in \mathbb{R}^d$ in such a way that the length of $\mathfrak{g}$ equals the length of the geodesic between $\mathfrak{g}$ and $\boldsymbol{p}$. The inverse of the exponential map is the *logarithmic map* $\mathrm{Log} : \mathcal{M} \to \mathcal{T}_{\boldsymbol{p}}\mathcal{M}$. These distance-preserving maps between the linear tangent space and the manifold allow computations to be performed on the manifold indirectly; they provide a way to perform computations that involve end-effector poses. Note that the implementation of the mappings is manifold specific (see e.g. Table I in [19]).

In PbD, task models are often described in a Gaussian-based probability density function (pdf). The Gaussian distribution is popular due to theoretical and practical reasons. The Riemannian equivalent of the Gaussian distribution is often approximated by

$$\mathcal{N}_{\mathcal{M}}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{\Sigma}|}} e^{-\frac{1}{2}\,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})^{\top}\,\boldsymbol{\Sigma}^{-1}\,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})}, \quad (1)$$

because the exact solution is computationally impractical [21]. Here, $\boldsymbol{x}, \boldsymbol{\mu} \in \mathcal{M}$ and $\boldsymbol{\Sigma} \in \mathcal{S}^+$ (the group of positive definite matrices) are defined in the tangent space of the point $\boldsymbol{\mu}$. Note that the Euclidean difference $\boldsymbol{x} - \boldsymbol{\mu}$ in the Mahanalobis distance is replaced by the Logarithmic map $\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})$.

Akin the Gaussian distribution, MLE, product and conditioning operations exists for the Riemannian Gaussian. We exploit these properties in our shared control methods to estimate the parameters of the Gaussian from demonstration data (MLE), combine statistical information from different sources (product), and infer the system state (conditioning). The result of these operations is approximated by a Riemannian Gaussian. Its center is found through a Gaussian-Newton optimization whose update rule is given by

$$\boldsymbol{\mu}_{k+1} \leftarrow \mathrm{Exp}_{\boldsymbol{\mu}_k}(\boldsymbol{\Delta})\,. \quad (2)$$

After convergence the covariance can be computed. Table I provides an overview of the formula required to compute $\boldsymbol{\Delta}$ and $\boldsymbol{\Sigma}$ for the operations mentioned.

We approximate more complex distributions using a mixture of Riemannian Gaussians

$$\mathcal{P}(\boldsymbol{x}) = \sum_{i=1}^{K} \pi_i \mathcal{N}_{\mathcal{M}}(\boldsymbol{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\,, \quad (3)$$

where π_i are the priors, with $\sum_i^K \pi_i = 1$. The parameters of this Gaussian Mixture Model (GMM) can be estimated using Expectation Maximization (EM), an iterative process in which the data are given weights for each cluster (Expectation step),

	Δ	$\boldsymbol{\Sigma}$
MLE	$\frac{1}{\sum_i^N h_i} \sum_{n=1}^{N} h_i \operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_n)$	$\frac{1}{\sum_i^N h_i} \sum_{n=1}^{N} h_i \operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_n) \operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_n)^\top$
Product	$\left(\sum_{p=1}^{P} \boldsymbol{\Sigma}_{\parallel p}^{-1}\right)^{-1} \sum_{p=1}^{P} \boldsymbol{\Sigma}_{\parallel p}^{-1} \operatorname{Log}_{\boldsymbol{\mu}}(\boldsymbol{\mu}_p)$	$\left(\sum_{p=1}^{P} \boldsymbol{\Sigma}_{\parallel p}^{-1}\right)^{-1}$
Condition	$\operatorname{Log}_{\boldsymbol{x}_\mathcal{O}}(\boldsymbol{\mu}_\mathcal{O}) + \boldsymbol{\Lambda}_{\parallel \mathcal{O}\mathcal{O}}^{-1} \boldsymbol{\Lambda}_{\parallel \mathcal{O}\mathcal{I}}^{\top} \operatorname{Log}_{\boldsymbol{x}_\mathcal{I}}(\boldsymbol{\mu}_\mathcal{I})$	$\boldsymbol{\Lambda}_{\parallel \mathcal{O}\mathcal{O}}^{-1}$

TABLE I: Overview of the equations required for Gaussian MLE, conditioning and product.

and the clusters are subsequently updated using a weighted MLE (Maximization step). We refer to [22] for a detailed description of EM for Riemannian GMMs.

Statistical inference of GMM is done efficiently using GMR. Similar to conditioning, this requires an iterative procedure to handle the potential non-linearities of the Riemannian manifold. The mean is computed as

$$\Delta = \sum_{i=1}^{K} h_i \operatorname{Log}_{\hat{\boldsymbol{\mu}}_\mathcal{O}}(\mathbb{E}[\mathcal{N}(\boldsymbol{x}_\mathcal{O} | \boldsymbol{x}_\mathcal{I}; \boldsymbol{\mu}_{\mathrm{i}}, \boldsymbol{\Sigma}_{\mathrm{i}})]), \tag{4}$$

$$\hat{\boldsymbol{\mu}}_\mathcal{O} \leftarrow \operatorname{Exp}_{\hat{\boldsymbol{\mu}}_\mathcal{O}}(\Delta). \tag{5}$$

and the covariance is computed after convergence

$$\hat{\boldsymbol{\Sigma}}_\mathcal{O} = \sum_{i}^{K} h_i \ \boldsymbol{\Sigma}_{\parallel i}, \ \text{where} \tag{6}$$

$$\boldsymbol{\Sigma}_{\parallel i} = \boldsymbol{\mathcal{A}}_{\parallel \boldsymbol{\mu}_i}^{\hat{\boldsymbol{\mu}}_\mathcal{O}}(\boldsymbol{L}_i)^\top \boldsymbol{\mathcal{A}}_{\parallel \boldsymbol{\mu}_i}^{\hat{\boldsymbol{\mu}}_\mathcal{O}}(\boldsymbol{L}_i), \tag{7}$$

and $\boldsymbol{\Sigma}_i = \boldsymbol{L}_i \boldsymbol{L}_i^\top$. Where $\boldsymbol{\mathcal{A}}_{\parallel \boldsymbol{a}}^{\boldsymbol{b}}()$ is an operator that parallel transports vectors from the tangent space at $\boldsymbol{a}$ to the tangent space at $\boldsymbol{b}$ (see [19] for details).

In this work, we employ LQR to generate haptic feedback on the end-effector orientation. The introduction of LQR simplified the design of optimal controllers for linear dynamical systems. It provides a way to control gains of a state feedback controller by defining a state and control cost matrix. In [20], we showed how the state cost can be obtained through PbD by relating it to the correlation information observed in the demonstration data. Even though robot poses are elements of a Riemannian manifold, the correlation information is described in the Euclidean tangent space. Therefore, we can still define a quadratic cost-function

$$c = \frac{1}{2} \int \left(\operatorname{Log}_{\bar{\boldsymbol{\mu}}}(\bar{\boldsymbol{p}})^\top \boldsymbol{Q} \ \operatorname{Log}_{\bar{\boldsymbol{\mu}}}(\bar{\boldsymbol{p}}) \ + \ \mathbf{u}^\top \boldsymbol{R} \mathbf{u} \right) \mathrm{d}t, \tag{8}$$

with state cost matrix $\boldsymbol{Q}$ and control cost matrix $\boldsymbol{R}$. $\bar{\boldsymbol{p}} = (\boldsymbol{p}, \dot{\boldsymbol{p}})$ and $\bar{\boldsymbol{\mu}} = (\boldsymbol{\mu}, \mathbf{0})$ are the current and desired state augmented with their first derivatives.

Furthermore, we can also define a linearized system at the setpoint of the LQR, namely

$$\begin{bmatrix} \dot{\boldsymbol{\epsilon}} \\ \ddot{\boldsymbol{\epsilon}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ \mathbf{0} & -\boldsymbol{M}^{-1}\boldsymbol{C} \end{bmatrix} \begin{bmatrix} \boldsymbol{\epsilon} \\ \dot{\boldsymbol{\epsilon}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \boldsymbol{M}^{-1} \end{bmatrix}. \tag{9}$$

with inertia matrix $\boldsymbol{M}$ and damping matrix $\boldsymbol{C}$, and $\boldsymbol{\epsilon} = \operatorname{Log}_{\boldsymbol{p_d}}(\boldsymbol{p})$. The controller minimizing the cost-function (8) subject to the liner time-invariant system (9) is of the form

$$\mathbf{u} \ = \ \boldsymbol{L} \operatorname{Log}_{\bar{\boldsymbol{\mu}}}(\bar{\boldsymbol{p}}). \tag{10}$$

The gain matrix $\boldsymbol{L}$ can be determined by solving an Algebraic Ricatti Equation, as in standard LQR [23].

III. Programming Shared Control by Demonstration

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* [8]—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:

$$\mathcal{N}_H = \mathcal{N}_H(\boldsymbol{\mu}_H, \boldsymbol{\Sigma}_H), \tag{11}$$

$$\mathcal{N}_A = \mathcal{N}_A(\boldsymbol{\mu}_A, \boldsymbol{\Sigma}_A). \tag{12}$$

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 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 the assistive system.

The remainder of this section describes the proposed method in detail. Section III-A discusses how the assistive system can be programmed by demonstration. Then, Section III-B 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 III-C.

A. Programming skill models by Demonstration

Instead of pre-defining the behavior of the assistive system, we program it by demonstration. We start by collecting a set of example motions. 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 II. 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. 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 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).

B. Shared Control Strategies

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 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.

1) State Shared Control (SSC): SSC combines the inputs after the interface on which the human operates. As depicted in Figure 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.

2) Haptic Shared Control (HSC): HSC combines the inputs at the user interface, as depicted in Figure 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 LQR. This is achieved using the cost function

$$c = \int (\boldsymbol{x}_s - \boldsymbol{\mu}_A)^\top \boldsymbol{\Sigma}_A^{-1} (\boldsymbol{x}_s - \boldsymbol{\mu}_A) + \boldsymbol{u}_A^\top \boldsymbol{R}_H \boldsymbol{u}_A \qquad (13)$$

with $\boldsymbol{x}_s$ the current state of the control interface, $\boldsymbol{u}$ the control input applied to the haptic interface, and $\boldsymbol{R}_H$ 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

$$\boldsymbol{u}_A = -\boldsymbol{L}(\boldsymbol{\mu}_A - \boldsymbol{x}_s). \qquad (14)$$

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.

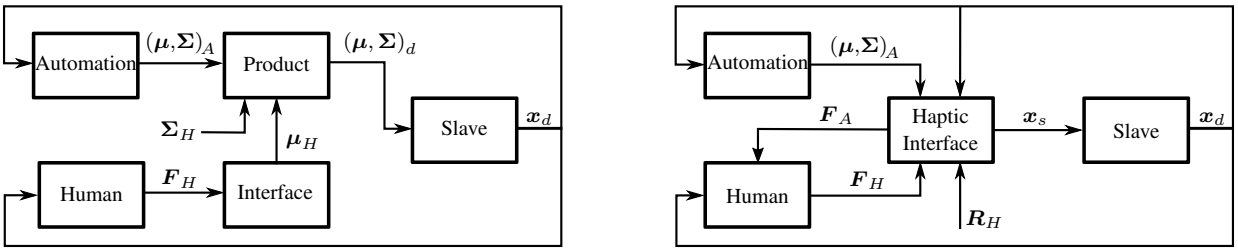
The level of automation results from the combination of tracking cost $\boldsymbol{\Sigma}^{-1}$ (automation confidence) and control cost $\boldsymbol{R}_H$ (human confidence), as these values determine the haptic feedback forces. When these forces are such that the human cannot resist them, the system acts autonomously. Manual operation, on the other hand, is achieved when the operator cannot detect the forces. This system thus generates a continuous range of autonomy levels.

3) Illustrative Comparison: We illustrate the differences between SSC and HSC using Figure 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 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 our previous work [20]. Such information is not available for HSC.

C. Confidence structures

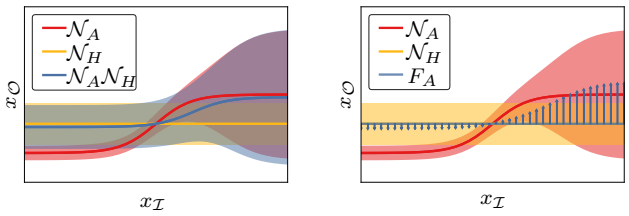
The covariance matrix supports a rich expression of confidence. Yet, this support comes at a cost: the amount of parameters to specify. By constraining the covariance structure, the number of parameters that needs to be specified can be reduced.

Figure 3 illustrates how a variety of mixing behaviors can be achieved using different covariance structures. Isotropic



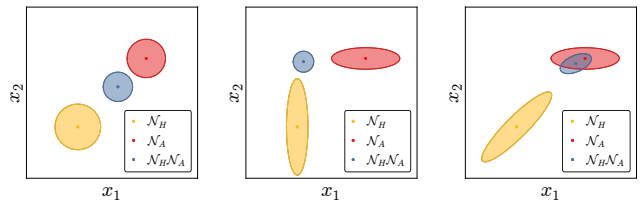
(a) State Shared Control (SSC) (b) Haptic Shared Control (HSC)

Fig. 1: Block diagrams visualizing the shared control strategies considered in this work.



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

Fig. 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 III-B for further description.



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

Fig. 3: Input mixing using different covariance structures (a–b). The control input of Human operator (H) and autonomous agent (A) are visualized by the yellow and red Gaussians (ellipsoids), respectively. The mixed control input is visualized by the blue Gaussian. The ellipsoid center indicates the desired state, and its contour the covariance.

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.* [24]. Note that the illustrations are based on SSC (product of Gaussians). In HSC, the structure of the control cost matrix $\boldsymbol{R}_H$ 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.

IV. Experimental Evaluation

A. Scenario

The Large Hadron Collider (LHC) at CERN in Switzerland is a hazardous environment. Radiation inside the LHC poses a health thread 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.

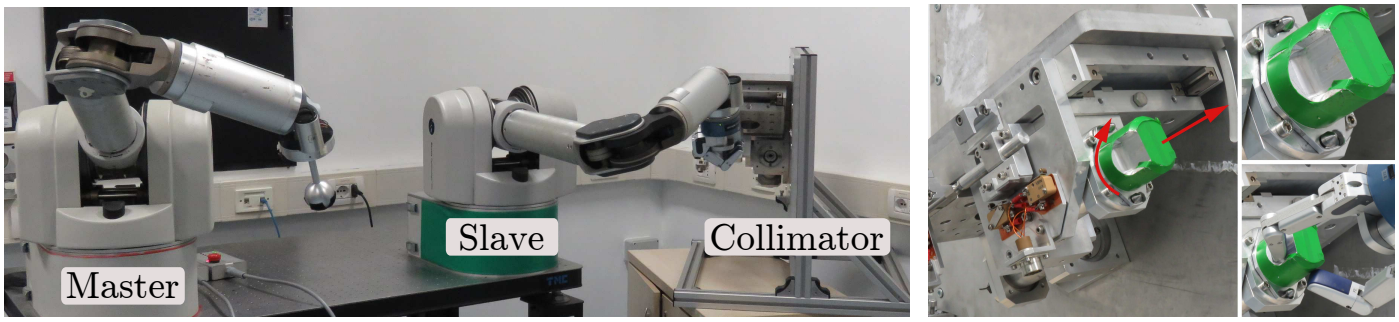
The experimental task is part of the maintenance procedure of a *collimator*—a device used to parallelize particle beams. The LHC can contain up to 152 collimators, which are located in radioactive areas [25]. This maintenance procedure involves the removal and replacement of a protection cover. Removal 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 Fig. 4b).

B. Experimental setup

The experimental setup, visualized in Figure 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.

C. 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



(a) Setup (b) Task

Fig. 4: Visualization of experimental setup. Refer to Sec. IV for a detailed description.

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 $\mathcal{P}(\boldsymbol{q}|\boldsymbol{x})$; 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 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.

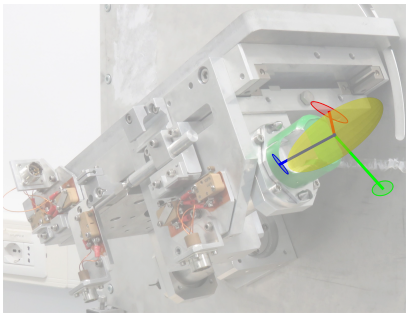


Fig. 5: Visualization of the trained skill on the collimator. The bound of the yellow ellipsoid indicates one standard deviation of the position covariance. The ellipsoid center indicates the mean. The orthogonal colored lines indicate the mean orientation. The colored ellipsoids at the end of each axis indicate 4 standard deviation of the rotational covariance.

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.

D. Results

1) Probabilistic Skill Model: We demonstrated the task on the collimator using kinesthetic teaching (see Figure 4b). The obtained model is visualized in Figure 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.

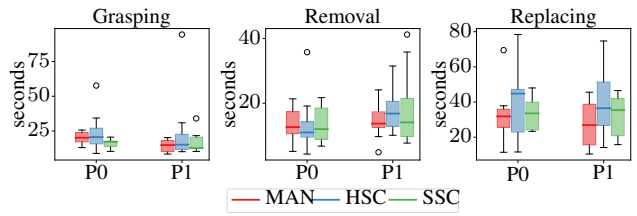


Fig. 6: Summary of the phase durations for the three teleoperation conditions, evaluated at two locations of the collimator.

	GRASPING	REMOVAL	REPLACING
MAN $\neq$ HSC	0.62, 0.21	0.47, 0.16	0.69, 0.24
MAN $\neq$ SSC	0.06, 0.25	0.64, 0.40	0.63, 0.75

TABLE II: 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).

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. $\mathcal{P}(\boldsymbol{q}|\boldsymbol{x}) \sim \mathcal{N}\left(\boldsymbol{\mu}^{\boldsymbol{q}|\boldsymbol{x}}, \boldsymbol{\Sigma}^{\boldsymbol{q}|\boldsymbol{x}}\right)$. This distribution forms the input of the assistive system.

2) 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 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 II, and demonstrate that there exist no significant differences ($p < 0.05$) in phase duration among the control methods.

Table III 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 III: Outcome of subjective evaluation. See Section IV-D for discussion of the results.

V. DISCUSSION

The presented work shares similarities with [10]. Nonetheless, our methods are different. Namely, both with SSC and HSC, we perform Gaussian regression from position to orientation while the work in [10] activates a virtual fixture 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 used in reproduction, providing a varying level of autonomy. Although most of the subjects indicated that they prefer a form of shared control while performing teleoperation, our subjective evaluation does not show that either HSC or SSC increases the teleoperation performance for our chosen measure. As this contradicts previous work on shared control [2], [4], [8], we discuss potential sources that could have caused this contradiction.

The task on which we evaluated the shared controllers was found difficult by our subjects. We found that users experienced difficulty removing and placing the cap over the two lock pins. This procedure is eased by a proper orientation of the end-effector, as provided by the shared controllers. Yet, additional fine manipulation was still required to remove or place the cap. In case the cap collides with the environment during the replacement, it slips 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 modifying the environment, but such modifications make application less realistic.

Finally, 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 easy 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 experiences. Using the current settings the subjects indicate they had to fight the forces provided by HSC (see Table III HSC.B). While the subjects disagreed this was the case for SSC (see Table III SSC.B), it indicates that thorough studies investigating the level of automation a user desires are further required.

In this work, we manually set the confidence on the user input. In practice, the teleoperator would do this via the control interface. Usmani *et al.* [26] take a different approach, and let the shared-control system judge the confidence of the user

input. The input confidence is based on the user Field of View (FoV): the user will receive low confidence on the control of state variables that cannot be properly estimated by the user, given its FoV. Although this provides an interesting extension to our approach, it also raises a fundamental question: who has the final say in control authority, the user or the assistive system?

In our approach the assistance is programmed offline: the demonstrations are collected and modeled during a training phase, and assistance is provided during a reproduction phase. Online programming of assistance would be an interesting extension. In this direction, Peternel *et al.* [27] proposed a Locally Weighted Regression (LWR) approach for SSC with binary control authority assignment. Extensions that consider continuous authority assignment and haptic feedback (HSC) are considered valuable but challenging, as it couples two adaptive systems (human and assistance).

VI. Conclusion

In this work, we proposed methods to generate shared control strategies for two different types of shared control, namely: SSC and HSC. We showed how current techniques for PbD can be used to generate shared control strategies that involve end-effector orientation.

We evaluated the approach on a realistic teleoperation scenario through a user study. The outcome of this study suggests that users prefer the learned shared control strategy over manual control. However, they were not unanimous in the preferred form of shared control. Our objective evaluation was not able to demonstrate that the learned shared control strategies decreased execution time of the teleoperation task. The use of an alternative interface, additional metrics, or a more constrained experimental environment are considered as ways of generating more distinctive results.

Besides extending the experimental evaluation, we aim at extending the developed method. In our current approach we manually set the input/output relation used in the regression, i.e. we manually decided to infer the desired orientation based on the position. Instead, determining this from data could result in discovering synergies (coupling movement relations) that are not easily perceivable and could have been overlooked when designed manually. Future work will focus on automatic selection of these causal relations.

References

[1] 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.

[2] 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.

[3] 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.

[4] L. B. Rosenberg, "Virtual fixtures: Perceptual tools for telerobotic manipulation," in *Proc. IEEE Intl Symp. Virtual Reality (VRAIS)*, Sep 1993, pp. 76–82.

[5] 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.

[6] 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 tele-manipulation: Effects of inaccuracies in guidance on task execution," *IEEE transactions on haptics*, vol. 8, no. 2, pp. 164–175, 2015.

[7] 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.

[8] 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.

[9] 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.

[10] 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)*, Sept 2015, pp. 7–13.

[11] 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.

[12] 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.

[13] 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.

[14] A. Dragan and S. Srinivasa, "A policy blending formalism for shared control," *International Journal of Robotics Research*, May 2013.

[15] 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.

[16] I. Havoutis and S. Calinon, "Learning assistive teleoperation behaviors from demonstration," in *Proc. IEEE Intl Symp. on Safety, Security and Rescue Robotics (SSRR)*, Lausanne, Switzerland, October 2016, pp. 258–263.

[17] 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.

[18] A. Pervez, A. Ali, J. H. Ryu, and D. Lee, "Novel learning from demonstration approach for repetitive teleoperation tasks," in *2017 IEEE World Haptics Conference (WHC)*, June 2017, pp. 60–65.

[19] 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.

[20] M. J. A. Zeestraten, I. Havoutis, S. Calinon, and D. G. Caldwell, "Learning task-space synergies using riemannian geometry," in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Vancouver, Canada, September 2017, pp. 73–78.

[21] 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.

[22] 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.

[23] A. E. Bryson, *Dynamic optimization*. Addison Wesley Longman, 1999.

[24] 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.

[25] 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," Tech. Rep., 2006.

[26] N. A. Usmani, T. H. Kim, and J. H. Ryu, "Dynamic authority distribution for cooperative teleoperation," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2015, pp. 5222–5227.

[27] L. Peternel, E. Oztop, and J. Babič, "A shared control method for online human-in-the-loop robot learning based on locally weighted regression," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 3900–3906.