Generalized Orientation Learning in Robot Task Space

Yanlong Huang, Fares J. Abu-Dakka, João Silvério, and Darwin G. Caldwell

Abstract— **In the context of imitation learning, several approaches have been developed so as to transfer human skills to robots, with demonstrations often represented in Cartesian or joint space. While learning Cartesian positions suffices for many applications, the end-effector orientation is required in many others. However, several crucial issues arising from learning orientations have not been adequately addressed yet. For instance, how can demonstrated orientations be adapted to pass through arbitrary desired points that comprise orientations and angular velocities? In this paper, we propose an approach that is capable of learning multiple orientation trajectories and adapting learned orientation skills to new situations (e.g., via-point and end-point), where both orientation and angular velocity are addressed. Specifically, we introduce a kernelized treatment to alleviate explicit basis functions when learning orientations. Several examples including comparison with the state-of-the-art dynamic movement primitives are provided to verify the effectiveness of our method.**

I. Introduction

In many complicated tasks (e.g., bi-manual manipulation [1] and robot table tennis [2], [3]), it is typically hard to manually define proper trajectories for robots beforehand, hence imitation learning is suggested in order to easily transfer human skills to robots [4]. The basic idea of imitation learning is to model movement patterns that underlie human skills and subsequently employ these patterns in new situations. Many results of imitation learning have been reported in the past few years, such as dynamic movement primitives (DMP) [5], probabilistic movement primitives (ProMP) [6], task-parameterized Gaussian mixture model (TP-GMM) [7], [8] and kernelized movement primitives (KMP) [9].

While the aforementioned skill learning approaches have been proven effective in robot trajectory generation [10], [11] (i.e., Cartesian and joint positions), learning of orientation in task space still imposes great challenges. Unlike position operations in Euclidean space, orientation is accompanied by additional constraints, e.g., the unit norm of the quaternion representation or the orthogonal constraint of rotation matrices. In many previous work, quaternion trajectories are learned and adapted via skill learning approaches (e.g., TP-GMM [1] and DMP [12]) without considering the unit norm constraint, leading to improper quaternions and hence requiring an additional renormalization.

Instead of learning quaternions in Euclidean space, a few approaches that comply with orientation constraints have been proposed. One recent approach is built on DMP [13], [14], where quaternions were used to represent orientation and a reformulation of DMP was developed to ensure proper quaternions over the course of orientation adaptation. However, [13], [14] can only adapt quaternions towards a new target with zero angular velocity due to the spring-damper dynamics that is inherited from the original DMP. Similar issue also arises in the contracting dynamics model [15]. Another solution of learning orientation was proposed in [16], where GMM was employed to model the distribution of quaternion displacements so as to avoid the quaternion constraint. However, this approach only focuses on orientation reproduction without addressing the adaptation issue. In contrast to [16] that learns the quaternion displacement, the Riemannian topology of the $\mathbb{S}^3$ manifold was exploited in [17] to probabilistically encode and reproduce distributions of quaternions. Moreover, [17] provides an extension to task-parameterized movements, which allows for adapting orientation tasks to different initial and final orientations. However, adaptation to orientation via-points and angular velocities is not provided.

If we consider the problem of adapting quaternions and angular velocities to pass through arbitrary desired points (e.g., via-point and end-point), no previous work in the scope of imitation learning (e.g., [1], [12], [13], [14], [15], [16], [17]) provides an all-encompassing solution. In addition, as suggested in [7], [18], probability distributions of multiple demonstrations could facilitate learning of important motion features and the design of optimal controllers [19], [20], [21]. However, due to the orientation constraint, it is not straightforward to model distributions of orientations, unlike Cartesian and joint positions. In this paper, we aim at providing a solution that is capable of learning multiple quaternion trajectories (Section II) while allowing for adaptations towards arbitrary desired points that consist of quaternions and angular velocities (Section III). For the purpose of clear comparison, the main contributions of the state-of-the-art approaches and our approach are summarized in Table I.

II. Learning Probabilistic Orientation Trajectories

As an effective way to represent orientation in task space, quaternions have been studied extensively, e.g., [1], [13], [14], [15], [16], [17]. However, due to the unit norm constraint, the direct probabilistic modeling of quaternion trajectories becomes intractable. Inspired by [13], [16], [17], we propose to transform quaternions into Euclidean space, which hence allows for the probabilistic modeling of transformed trajectories (Section II-A). Subsequently, we exploit the distribution of transformed trajectories using a kernelized

All authors are with Department of Advanced Robotics, Istituto Italiano di Tecnologia, Via Morego 30, 16163 Genoa, Italy, firstname.lastname@iit.it

This work was supported by the Italian Ministry of Defense.

TABLE I

Comparison Among the State-of-the-Art and Our Approach

	Probabilistic	Unit-norm	Via-quaternion	Via-angular velocity	End-quaternion	End-angular velocity
Silvério *et al.* [1]	✓	-	-	-	✓	-
Pastor *et al.* [12]	-	-	-	-	✓	-*
Ude *et al.* [13], Abu-dakka *et al.* [14]	-	✓	-	-	✓	-*
Ravichandar et al. [15]	✓	✓	-	-	✓	-*
Kim *et al.* [16]	✓	✓	-	-	-	-
Zeestraten *et al.* [17]	✓	✓	-	-	✓	-
Our approach	✓	✓	✓	✓	✓	✓

* In these works, primitives end with zero angular velocity, i.e., one can not set a desired non-zero velocity.

approach based on KMP [9], whose predictions allow for the retrieval of proper quaternions (Section II-B).

A. Probabilistic Modeling of Quaternion Trajectories

A straightforward way of modeling quaternions is to transform them into Euclidean space, as done in [13], [16], [17]. For the sake of clarity, let us define quaternions $\mathbf{q}_1 = v_1 + \mathbf{u}_1$ and $\mathbf{q}_2 = v_2 + \mathbf{u}_2$, where $\mathbf{q}_i \in \mathbb{S}^3$, $v_i \in \mathbb{R}$ and $\mathbf{u}_i \in \mathbb{R}^3$, $i \in \{1, 2\}$. Besides, we write $\bar{\mathbf{q}}_2 = v_2 - \mathbf{u}_2$ as the conjugation of $\mathbf{q}_2$ and, $\mathbf{q} = \mathbf{q}_1 * \bar{\mathbf{q}}_2 = v + \mathbf{u}$ as the quaternion product[1] of $\mathbf{q}_1$ and $\bar{\mathbf{q}}_2$. The function $\log(\cdot) : \mathbb{S}^3 \mapsto \mathbb{R}^3$ that can be used to determine the difference vector between $\mathbf{q}_1$ and $\mathbf{q}_2$ is defined as [13]

$$\log(\mathbf{q}_1 * \bar{\mathbf{q}}_2) = \log(\mathbf{q}) = \begin{cases} \arccos(v)\dfrac{\mathbf{u}}{||\mathbf{u}||}, \mathbf{u} \neq \mathbf{0} \\ [0 \;\; 0 \;\; 0]^{\top}, \qquad otherwise. \end{cases} \tag{1}$$

With this function, demonstrated quaternions can be projected into Euclidean space.

Assuming that we can access a set of demonstrations $\mathbf{D}_q = \{\{t_{n,m}, \mathbf{q}_{n,m}\}_{n=1}^{N}\}_{m=1}^{M}$ with N being the time length and M the number of demonstrations, where $\mathbf{q}_{n,m}$ denotes a quaternion at the n-th time-step from the m-th demonstration. Note that two quaternions are needed in (1) so as to carry out the difference operation, we introduce an auxiliary quaternion $\mathbf{q}_a$, which is subsequently used for transforming demonstrated quaternions into Euclidean space, yielding new trajectories as $\mathbf{D}_\zeta = \{\{t_{n,m}, \boldsymbol{\zeta}_{n,m}, \dot{\boldsymbol{\zeta}}_{n,m}\}_{n=1}^{N}\}_{m=1}^{M}$ with

$$\boldsymbol{\zeta}_{n,m} = \log(\mathbf{q}_{n,m} * \bar{\mathbf{q}}_a) \tag{2}$$

and $\dot{\boldsymbol{\zeta}}_{n,m} \in \mathbb{R}^3$ being the derivative of $\boldsymbol{\zeta}_{n,m} \in \mathbb{R}^3$. Please note that the new trajectories $\mathbf{D}_\zeta$ can be used to recover quaternion trajectories $\mathbf{D}_q$, as explained in Section II-B. For the purpose of simplicity, we denote $\boldsymbol{\eta} = [\boldsymbol{\zeta}^{\top} \; \dot{\boldsymbol{\zeta}}^{\top}]^{\top}$ and accordingly $\mathbf{D}_\zeta$ becomes $\mathbf{D}_\eta = \{\{t_{n,m}, \boldsymbol{\eta}_{n,m}\}_{n=1}^{N}\}_{m=1}^{M}$.

From now on, we can apply various probabilistic modeling approaches to new trajectories $\mathbf{D}_\eta$. To take GMM as an example [7], the joint probability distribution $\mathcal{P}(t, \boldsymbol{\eta})$ can be estimated through expectation-maximization algorithm, leading to

$$\mathcal{P}(t, \boldsymbol{\eta}) \sim \sum_{c=1}^{C} \pi_c \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c), \tag{3}$$

where π_c denotes prior probability of the c-th Gaussian component whose mean and covariance are, respectively, $\boldsymbol{\mu}_c = \begin{bmatrix} \boldsymbol{\mu}_{t,c} \\ \boldsymbol{\mu}_{\eta,c} \end{bmatrix}$ and $\boldsymbol{\Sigma}_c = \begin{bmatrix} \boldsymbol{\Sigma}_{tt,c} & \boldsymbol{\Sigma}_{t\eta,c} \\ \boldsymbol{\Sigma}_{\eta t,c} & \boldsymbol{\Sigma}_{\eta\eta,c} \end{bmatrix}$[2]. Furthermore, Gaussian mixture regression (GMR) [7], [22] can be utilized to retrieve the conditional probability distribution

$$\mathcal{P}(\boldsymbol{\eta}|t) = \sum_{c=1}^{C} h_c(t) \mathcal{N}(\bar{\boldsymbol{\mu}}_c(t), \bar{\boldsymbol{\Sigma}}_c) \tag{4}$$

with $h_c(t) = \frac{\pi_c \mathcal{N}(t|\boldsymbol{\mu}_{t,c}, \boldsymbol{\Sigma}_{tt,c})}{\sum_{i=1}^{C} \pi_i \mathcal{N}(t|\boldsymbol{\mu}_{t,i}, \boldsymbol{\Sigma}_{tt,i})}$, $\bar{\boldsymbol{\mu}}_c(t) = \boldsymbol{\mu}_{\eta,c} + \boldsymbol{\Sigma}_{\eta t,c} \boldsymbol{\Sigma}_{tt,c}^{-1}(t - \boldsymbol{\mu}_{t,c})$ and $\bar{\boldsymbol{\Sigma}}_c = \boldsymbol{\Sigma}_{\eta\eta,c} - \boldsymbol{\Sigma}_{\eta t,c} \boldsymbol{\Sigma}_{tt,c}^{-1} \boldsymbol{\Sigma}_{t\eta,c}$. Note that the result in (4) can be approximated by a single Gaussian, i.e.,

$$\mathcal{P}(\boldsymbol{\eta}|t) = \mathcal{N}(\hat{\boldsymbol{\mu}}_t, \hat{\boldsymbol{\Sigma}}_t) \tag{5}$$

with $\hat{\boldsymbol{\mu}}_t = h_c(t)\bar{\boldsymbol{\mu}}_c(t)$ and $\hat{\boldsymbol{\Sigma}}_t = h_c(t)\left(\bar{\boldsymbol{\mu}}_c(t)\bar{\boldsymbol{\mu}}_c^{\top}(t) + \bar{\boldsymbol{\Sigma}}_c\right) - \hat{\boldsymbol{\mu}}_t \hat{\boldsymbol{\mu}}_t^{\top}$, Please refer to [7], [9], [22] for more details. Therefore, for a given time sequence $\{t_n\}_{n=1}^{N}$, a probabilistic reference trajectory $\mathbf{D}_r = \{t_n, \hat{\boldsymbol{\mu}}_n, \hat{\boldsymbol{\Sigma}}_n\}_{n=1}^{N}$ can be obtained. Note that $\mathbf{D}_r$ can be viewed as representative of $\mathbf{D}_\eta$ since it encapsulates the distribution of trajectories in $\mathbf{D}_\eta$ in terms of mean and covariance, and hence we exploit $\mathbf{D}_r$ instead in the next subsection.

B. Learning Quaternions Using A Kernelized Approach

As done in KMP [9], [23], we first write $\boldsymbol{\eta}$ in a parameterized way[3], i.e.,

$$\boldsymbol{\eta}(t) = \left[\begin{array}{c} \boldsymbol{\zeta}(t) \\ \hdashline \dot{\boldsymbol{\zeta}}(t) \end{array}\right] = \boldsymbol{\Theta}^{\top}(t)\mathbf{w} = \left[\begin{array}{ccc} \boldsymbol{\phi}^{\top}(t) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\phi}^{\top}(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\phi}^{\top}(t) \\ \hdashline \dot{\boldsymbol{\phi}}^{\top}(t) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \dot{\boldsymbol{\phi}}^{\top}(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \dot{\boldsymbol{\phi}}^{\top}(t) \end{array}\right] \mathbf{w} \tag{6}$$

with $\boldsymbol{\phi}(t) \in \mathbb{R}^B$ being a B-dimensional basis function vector. Note that the parameter vector $\mathbf{w} \in \mathbb{R}^{3B}$ is unknown. In order to learn the probabilistic reference trajectories

[1] Quaternion product is defined as: $(v_1 + \mathbf{u}_1) * (v_2 + \mathbf{u}_2) = (v_1 v_2 - \mathbf{u}_1^{\top}\mathbf{u}_2) + (v_1\mathbf{u}_2 + v_2\mathbf{u}_1 + \mathbf{u}_1 \times \mathbf{u}_2)$.

[2] For the sake of notation consistency, we still use vector notations $\mathbf{u}_{t,c}$ and $\boldsymbol{\Sigma}_{tt,c}$ to represent scalars.

[3] Similar parametric strategies were used in DMP [5] and ProMP [6].

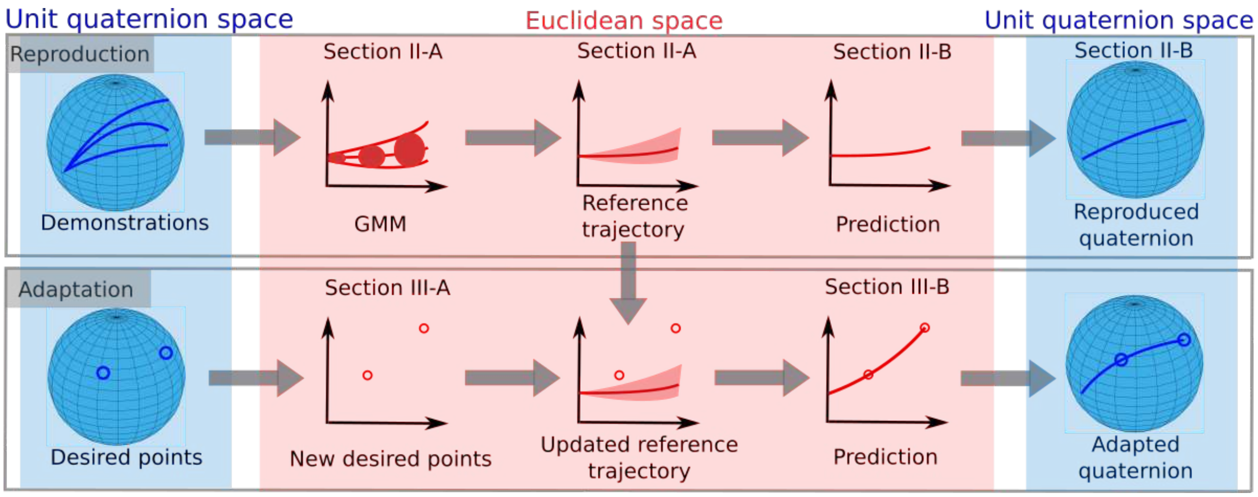


Fig. 1. Overview of quaternion reproduction and adaptation. *Top row*: given demonstrated quaternion trajectories, we first transform them into Euclidean space and model these obtained trajectories using GMM. Subsequently, we can extract a probabilistic reference trajectory by using GMR. Finally, we learn the reference trajectory using a kernelized learning approach and retrieve a trajectory (in Euclidean space) that is later used to recover a quaternion trajectory. *Bottom row*: Given desired quaternion states, we transform them into Euclidean space and, subsequently, concatenate new desired points with the reference trajectory (extracted from original demonstrations). Similarly to the reproduction case, we can generate the adapted trajectory in Euclidean space and recover its corresponding quaternion trajectory.

$\mathbf{D}_r$, we consider the problem of maximizing the posterior probability[4]

$$J(\mathbf{w}) = \prod_{n=1}^{N} \mathcal{P}(\boldsymbol{\Theta}^\top(t_n)\mathbf{w}|\hat{\boldsymbol{\mu}}_n, \hat{\boldsymbol{\Sigma}}_n). \tag{7}$$

It can be proved that the optimal solution $\mathbf{w}^*$ to (7) can be computed as

$$\mathbf{w}^* = \mathrm{argmin}_{\mathbf{w}} \sum_{n=1}^{N} (\boldsymbol{\Theta}^\top(t_n)\mathbf{w} - \hat{\boldsymbol{\mu}}_n)^\top (\hat{\boldsymbol{\Sigma}}_n)^{-1} (\boldsymbol{\Theta}^\top(t_n)\mathbf{w} - \hat{\boldsymbol{\mu}}_n), \tag{8}$$

where the objective to be minimized can be viewed as the sum of covariance-weighted squared errors.

Like kernel ridge regression [24], [25], [26], the optimal solution $\mathbf{w}^*$ of (8) can be computed. Thus, for an inquiry point t^*, its corresponding output $\boldsymbol{\eta}(t^*)$ can be predicted as (see [9] for details)

$$\boldsymbol{\eta}(t^*) = \boldsymbol{\Theta}(t^*)^\top \mathbf{w}^* = \boldsymbol{\Theta}(t^*)^\top \boldsymbol{\Phi}(\boldsymbol{\Phi}^\top \boldsymbol{\Phi} + \lambda \boldsymbol{\Sigma})^{-1} \boldsymbol{\mu} \tag{9}$$

where $\lambda > 0$ is a small constant and

$$\begin{aligned} \boldsymbol{\Phi} &= [\boldsymbol{\Theta}(t_1)\, \boldsymbol{\Theta}(t_2)\, \cdots\, \boldsymbol{\Theta}(t_N)], \\ \boldsymbol{\Sigma} &= blockdiag(\hat{\boldsymbol{\Sigma}}_1, \hat{\boldsymbol{\Sigma}}_2, \ldots, \hat{\boldsymbol{\Sigma}}_N), \\ \boldsymbol{\mu} &= [\hat{\boldsymbol{\mu}}_1^\top\, \hat{\boldsymbol{\mu}}_2^\top\, \cdots\, \hat{\boldsymbol{\mu}}_N^\top]^\top. \end{aligned} \tag{10}$$

Furthermore, (9) can be kernelized as

$$\boldsymbol{\eta}(t^*) = \mathbf{k}^*(\mathbf{K} + \lambda \boldsymbol{\Sigma})^{-1} \boldsymbol{\mu} \tag{11}$$

with $\mathbf{k}^*_{[i]} = \mathbf{k}(t^*, t_i)$ and $\mathbf{K}_{[i,j]} = \mathbf{k}(t_i, t_j)$, $i \in \{1, 2, \ldots, N\}, j \in \{1, 2, \ldots, N\}$, where $\mathbf{k}^*_{[i]} \in \mathbb{R}^{6\times 6}$ denotes the block-component at the i-th column of $\mathbf{k}^* \in \mathbb{R}^{6\times 6N}$, $\mathbf{K}_{[i,j]} \in \mathbb{R}^{6\times 6}$ denotes the block-component at the i-th row and the j-th column of $\mathbf{K} \in \mathbb{R}^{6N\times 6N}$, $\mathbf{k}(\cdot,\cdot)$ is defined by

$$\mathbf{k}(t_i, t_j) = \boldsymbol{\Theta}(t_i)^\top \boldsymbol{\Theta}(t_j) = \begin{bmatrix} k_{t,t}(t_i, t_j)\mathbf{I}_3 & k_{t,d}(t_i, t_j)\mathbf{I}_3 \\ k_{d,t}(t_i, t_j)\mathbf{I}_3 & k_{d,d}(t_i, t_j)\mathbf{I}_3 \end{bmatrix} \tag{12}$$

with[5]

$$\begin{aligned} k_{t,t}(t_i, t_j) &= k(t_i, t_j), \\ k_{t,d}(t_i, t_j) &= \frac{k(t_i, t_j + \delta) - k(t_i, t_j)}{\delta}, \\ k_{d,t}(t_i, t_j) &= \frac{k(t_i + \delta, t_j) - k(t_i, t_j)}{\delta}, \\ k_{d,d}(t_i, t_j) &= \frac{k(t_i+\delta, t_j+\delta) - k(t_i+\delta, t_j) - k(t_i, t_j+\delta) + k(t_i, t_j)}{\delta^2}, \end{aligned}$$

where $\delta > 0$ is a very small constant and $k(t_i, t_j) = \boldsymbol{\phi}(t_i)^\top \boldsymbol{\phi}(t_j)$ represents the kernel function.

Let us recall that quaternion trajectories have been transformed into Euclidean space by using (1) (as explained in Section II-A). Thus, once we have determined $\boldsymbol{\eta}(t^*)$ at a query point t^* via (11), we can use its position component $\boldsymbol{\zeta}(t^*)$ to recover the corresponding quaternion $\mathbf{q}(t^*)$. Specifically, $\mathbf{q}(t^*)$ is determined by

$$\mathbf{q}(t^*) = \exp(\boldsymbol{\zeta}(t^*)) * \mathbf{q}_a, \tag{13}$$

where the function $\exp(\cdot) : \mathbb{R}^3 \mapsto \mathbb{S}^3$ is [13]

$$\exp(\boldsymbol{\zeta}) = \begin{cases} \cos(||\boldsymbol{\zeta}||) + \sin(||\boldsymbol{\zeta}||)\dfrac{\boldsymbol{\zeta}}{||\boldsymbol{\zeta}||}, & \boldsymbol{\zeta} \neq \mathbf{0} \\ 1 + [0\ \ 0\ \ 0]^\top, & otherwise. \end{cases} \tag{14}$$

An overview of learning quaternions is depicted in the top row of Fig. 1. So far, the developed approach is limited for orientation reproduction, we will show orientation adaptation in the next section, where quaternion and angular-velocity

[4] In contrast to KMP [9] that estimates the distribution of $\mathbf{w}$ from an information-theory perspective, we here follow the treatment in [23] where $\mathbf{w}$ is considered as an unknown but deterministic parameter vector.

[5] Note that $\dot{\boldsymbol{\phi}}(t)$ is approximated by $\dot{\boldsymbol{\phi}}(t) \approx \frac{\boldsymbol{\phi}(t+\delta) - \boldsymbol{\phi}(t)}{\delta}$ in order to facilitate the following kernelized treatment.

Algorithm 1 *Quaternion adaptations towards desired points*

```
Learn from demonstrations
    - Define $\mathbf{q}_a$ and collect demonstrated quaternions $\mathbf{D}_q = \{\{t_{n,m}, \mathbf{q}_{n,m}\}_{n=1}^{N}\}_{m=1}^{M}$
    - Transform $\mathbf{D}_q$ into Euclidean space via (2), yielding $\mathbf{D}_\eta = \{\{t_{n,m}, \boldsymbol{\eta}_{n,m}\}\}_{n=1}^{N}\}_{m=1}^{M}$
    - Model the joint distribution $\mathcal{P}(t,\boldsymbol{\eta})$ from $\mathbf{D}_\eta$ using (3)
    - Extract the probabilistic reference trajectory $\mathbf{D}_r = \{t_n, \hat{\boldsymbol{\mu}}_n, \hat{\boldsymbol{\Sigma}}_n\}_{n=1}^{N}$ via (5)
Update reference trajectory
    - Set desired quaternion states $\tilde{\mathbf{D}}_q = \{\tilde{t}_h, \tilde{\mathbf{q}}_h, \tilde{\boldsymbol{\omega}}_h\}_{h=1}^{H}$
    - Set covariances $\{\tilde{\boldsymbol{\Sigma}}_h\}_{h=1}^{H}$ for adaptation precisions
    - Transform $\tilde{\mathbf{D}}_q$ via (15) and (19), yielding an additional reference trajectory $\tilde{\mathbf{D}}_r = \{\tilde{t}_h, \tilde{\boldsymbol{\eta}}_h, \boldsymbol{\Sigma}_h\}_{h=1}^{H}$
    - Update $\mathbf{D}_r$ by concatenating $\mathbf{D}_r$ and $\tilde{\mathbf{D}}_r$
Predict adapted quaternions
    - Define $\lambda$ and $k(\cdot,\cdot)$
    Input: the query point $t^*$
    - Compute $\mathbf{k}^*$, $\mathbf{K}$, $\boldsymbol{\Sigma}$ and $\boldsymbol{\mu}$ using (10) and (12)
    - Predict $\boldsymbol{\eta}(t^*)$ through (11)
    - Predict $\mathbf{q}(t^*)$ using $\boldsymbol{\zeta}(t^*)$ through (13)
    Output: $\mathbf{q}(t^*)$
```

profiles can be modulated so as to pass through any desired points (e.g., via-/end- points).

III. Adaptation of Orientation Trajectories

Similarly to trajectory adaptation in terms of Cartesian and joint positions (and/or velocities) [5], [6], [7], [8], [9], the capability of adapting orientation in Cartesian space is also important for robots in many cases (e.g. bi-manual operations and pouring tasks). To take a pouring task as an example, the orientation of the bottle should be adapted according to the height of the cup. In this section, we consider the problem of adapting orientation trajectory in terms of desired quaternions and angular velocities. To do so, we propose to transform desired orientation states into Euclidean space (Section III-A), and subsequently we reformulate the kernelized learning approach to incorporate the transformed desired points (Section III-B). Finally, the adapted trajectory in Euclidean space can be used to retrieve its corresponding adapted quaternion trajectory. An overview of adapting quaternions is depicted in the bottom row of Fig. 1.

A. Transform Desired Quaternion States

Let us denote H desired quaternion states as $\tilde{\mathbf{D}}_q = \{\tilde{t}_h, \tilde{\mathbf{q}}_h, \tilde{\boldsymbol{\omega}}_h\}_{h=1}^{H}$, where $\tilde{\mathbf{q}}_h$ and $\tilde{\boldsymbol{\omega}}_h$ represent desired quaternion and angular velocity at time $\tilde{t}_h$, respectively. Since both the modeling operation (3)−(5) and the prediction operation (11) are carried out in Euclidean space, we need to transform desired quaternion states $\tilde{\mathbf{D}}_q$ into Euclidean space in order to facilitate adaptations of quaternion trajectories. Similarly to (2), the desired quaternion $\tilde{\mathbf{q}}_h$ can be transformed as

$$\tilde{\boldsymbol{\zeta}}_h = \log(\tilde{\mathbf{q}}_h * \bar{\mathbf{q}}_a). \tag{15}$$

However, in order to incorporate the desired angular velocity $\tilde{\boldsymbol{\omega}}_h$, we resort to the relationship between derivatives of quaternions and angular velocities, i.e., [13]

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\omega} * \mathbf{q} \Rightarrow \mathbf{q}(t + \delta_t) = \exp(\frac{\boldsymbol{\omega}}{2}\delta_t) * \mathbf{q}(t), \tag{16}$$

where $\delta_t > 0$ denotes a small constant. By using (16), we can compute the desired quaternion at time $\tilde{t}_h + \delta_t$ as

$$\tilde{\mathbf{q}}(\tilde{t}_h + \delta_t) = \exp(\frac{\tilde{\boldsymbol{\omega}}_h}{2}\delta_t) * \tilde{\mathbf{q}}_h, \tag{17}$$

which is subsequently transformed into Euclidean space via (2), resulting in

$$\begin{aligned} \tilde{\boldsymbol{\zeta}}(\tilde{t}_h + \delta_t) &= \log(\tilde{\mathbf{q}}(\tilde{t}_h + \delta_t) * \bar{\mathbf{q}}_a) \\ &= \log\left(\left(\exp(\frac{\tilde{\boldsymbol{\omega}}_h}{2}\delta_t) * \tilde{\mathbf{q}}_h\right) * \bar{\mathbf{q}}_a\right). \end{aligned} \tag{18}$$

Thus, we can approximate the derivative of $\tilde{\boldsymbol{\zeta}}_h$ as

$$\begin{aligned} \dot{\tilde{\boldsymbol{\zeta}}}_h &\approx \frac{\tilde{\boldsymbol{\zeta}}(\tilde{t}_h + \delta_t) - \tilde{\boldsymbol{\zeta}}_h}{\delta_t} \\ &= \frac{\log\left(\left(\exp(\frac{\tilde{\boldsymbol{\omega}}_h}{2}\delta_t) * \tilde{\mathbf{q}}_h\right) * \bar{\mathbf{q}}_a\right) - \log(\tilde{\mathbf{q}}_h * \bar{\mathbf{q}}_a)}{\delta_t}. \end{aligned} \tag{19}$$

Now, the desired quaternion states $\tilde{\mathbf{D}}_q$ can be transformed into $\tilde{\mathbf{D}}_\zeta = \{\tilde{t}_h, \tilde{\boldsymbol{\zeta}}_h, \dot{\tilde{\boldsymbol{\zeta}}}_h\}_{h=1}^{H}$ via (15) and (19), which can be further rewritten as $\tilde{\mathbf{D}}_\eta = \{\tilde{t}_h, \tilde{\boldsymbol{\eta}}_h\}_{h=1}^{H}$ with $\tilde{\boldsymbol{\eta}}_h = [\tilde{\boldsymbol{\zeta}}_h^\top \, \dot{\tilde{\boldsymbol{\zeta}}}_h^\top]^\top$. In addition, we can design a covariance $\tilde{\boldsymbol{\Sigma}}_h$ for each desired point $\tilde{\boldsymbol{\eta}}_h$ to address the precision of adaptations. Namely, a high or low precision can be enforced by a small or large covariance, respectively. Thus, we can obtain an additional probabilistic reference trajectory $\tilde{\mathbf{D}}_r = \{\tilde{t}_h, \tilde{\boldsymbol{\eta}}_h, \tilde{\boldsymbol{\Sigma}}_h\}_{h=1}^{H}$ to indicate the transformed desired quaternion states.

B. Adaptation of Quaternion Trajectories

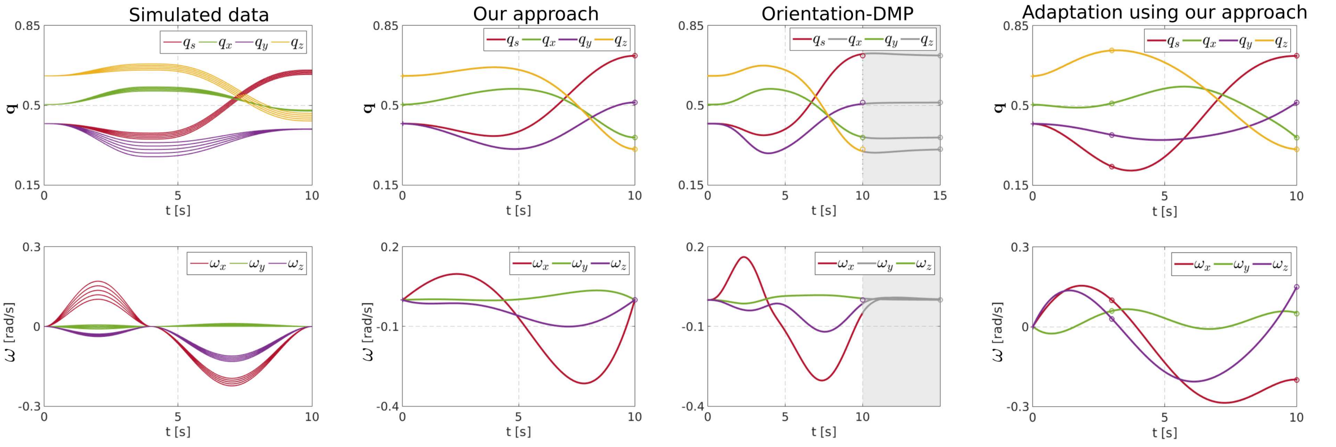
Following the adaptation scheme in KMP [9], we reformulate the objective in (8) so that the additional reference trajectory $\hat{\mathbf{D}}_r$ is incorporated, leading to a new objective

$$\begin{aligned} \mathbf{w}^* = \mathrm{argmin}_{\mathbf{w}} &\sum_{n=1}^{N} (\boldsymbol{\Theta}^\top(t_n)\mathbf{w} - \hat{\boldsymbol{\mu}}_n)^\top (\hat{\boldsymbol{\Sigma}}_n)^{-1} (\boldsymbol{\Theta}^\top(t_n)\mathbf{w} - \hat{\boldsymbol{\mu}}_n) \\ &+ \sum_{h=1}^{H} \left(\boldsymbol{\Theta}^\top(\tilde{t}_h)\mathbf{w} - \tilde{\boldsymbol{\eta}}_h\right)^\top (\tilde{\boldsymbol{\Sigma}}_h)^{-1} \left(\boldsymbol{\Theta}^\top(\tilde{t}_h)\mathbf{w} - \tilde{\boldsymbol{\eta}}_h\right), \end{aligned} \tag{20}$$

which can be further rewritten in a compact way

$$\mathbf{w}^* = \mathrm{argmin}_{\mathbf{w}} \sum_{l=1}^{N+H} (\boldsymbol{\Theta}^\top(\bar{t}_l)\mathbf{w} - \bar{\boldsymbol{\eta}}_l)^\top (\bar{\boldsymbol{\Sigma}}_l)^{-1} (\boldsymbol{\Theta}^\top(\bar{t}_l)\mathbf{w} - \bar{\boldsymbol{\eta}}_l) \tag{21}$$

with $\{\bar{t}_l = t_l, \bar{\boldsymbol{\eta}}_l = \hat{\boldsymbol{\mu}}_l, \bar{\boldsymbol{\Sigma}}_l = \hat{\boldsymbol{\Sigma}}_l\}, l \in \{1, 2, \ldots, N\}$ and $\{\bar{t}_l = \tilde{t}_{l-N}, \bar{\boldsymbol{\eta}}_l = \tilde{\boldsymbol{\eta}}_{l-N}, \bar{\boldsymbol{\Sigma}}_l = \tilde{\boldsymbol{\Sigma}}_{l-N}\}, l \in \{N+1, N+2, \ldots, N+H\}$. It can be observed that the new objective (21) shares the same form with (8), except that the reference trajectory in (21) is longer than that in (8), thus the solution of (21) can be determined in a similar way. Finally, $\boldsymbol{\eta}(t) = [\boldsymbol{\zeta}^\top(t) \, \dot{\boldsymbol{\zeta}}^\top(t)]^\top$ can be computed via (11) and, subsequently, $\mathbf{q}(t)$ is recovered from (13) by using $\boldsymbol{\zeta}(t)$. In this case, $\mathbf{q}(t)$



(a) (b1) (b2) (c)

Fig. 2. Evaluations of our approach on simulated examples. (*a*) shows simulated quaternion trajectories and their corresponding angular velocities. (*b*) displays adapted quaternion trajectories towards new target (i.e., end-point) and the corresponding angular velocities by using our approach (*b1*) and orientation-DMP [13], [14] (*b2*). Note that for both approaches the desired movement duration is 10*s*, the shaded area denotes extra time required for DMP. The circles with bright colors denote desired quaternions and angular velocities, while the gray circles in (*b2*) correspond to the delayed desired points. (*c*) depicts adapted quaternion and angular velocity profiles with various constraints of desired points.

TABLE II
PLANNED ERRORS OF OUR APPROACH AND ORIENTATION–DMP

	Quaternion distance error*		Angular-velocity error	
	$t = 10s^{**}$	$t = 15s$	$t = 10s$	$t = 15s$
Our approach	0	-	0.0017	-
Orientation–DMP	0.0285	0.0046	0.0513	0.0034

**Quaternion distance is calculated by* [13]:

$$d(\mathbf{q}_1, \mathbf{q}_2) = \begin{cases} 2\pi, & \mathbf{q}_1 * \bar{\mathbf{q}}_2 = -1 + [0\,0\,0]^\top \\ 2\|\log(\mathbf{q}_1 * \bar{\mathbf{q}}_2)\|, & \textit{otherwise}. \end{cases}$$

***Note that the desired movement duration is* 10*s*.

is capable of passing through the desired quaternions $\tilde{\mathbf{q}}_h$ with desired angular velocities $\tilde{\boldsymbol{\omega}}_h$ at time $\tilde{t}_h$. The entire approach of quaternion adaptations is summarized in Algorithm 1.

IV. EVALUATIONS

In this section, we report several examples to illustrate the performance of our approach: (*i*) orientation adaptation towards a desired target point (Section IV-A), where orientation-DMP [13], [14] is also employed as a comparison; (*ii*) orientation adaptation towards arbitrary desired points in terms of quaternions and angular velocities (Section IV-A); (*iii*) Concurrent adaptations of Cartesian position and orientation (Section IV-B), where a painting task on a real Barrett WAM robot is carried out.

A. Evaluations on Simulated Quaternion Trajectories

We collected five simulated quaternion trajectories with time-length $10s$ as depicted in Fig. 2(*a*), where minimal jerk polynomial and renormalization are used to generate smooth and proper quaternion trajectories. In order to show the performance of our approach, we first compare it with orientation–DMP [13], [14]. Since orientation-DMP can only address target adaptation (while having zero velocity at the ending point), we consider an example that merely requires orientation adaptation towards a new target (i.e., quaternion) with zero angular velocity. The desired point is defined as $\tilde{t}_1 = 10s$, $\tilde{\mathbf{q}}_1 = [0.7172\ \ 0.3586\ \ 0.5123\ \ 0.3074]$, $\tilde{\boldsymbol{\omega}}_1 = [0\ \ 0\ \ 0]$. The auxiliary quaternion $\mathbf{q}_a$ is set as the initial value of simulated quaternion trajectories. The Gaussian kernel $k(t_i, t_j) = \exp(-\ell(t_i - t_j)^2)$ with $\ell = 0.01$ and the regularized factor $\lambda = 1$ are used in our approach.

The adapted quaternion and angular-velocity profiles by using our approach and orientation–DMP are provided in Fig. 2(*b1*)-(*b2*). It can be seen from Fig. 2(*b1*) that our approach is capable of generalizing learned quaternion trajectories to the new target point $\tilde{\mathbf{q}}_1$ while having zero angular velocity at the ending time $\tilde{t}_1$. However, orientation–DMP needs extra time (i.e., depicted by shaded area) to converge to the desired point. The planned errors of both methods in comparison with the desired point is summarized in Table II, showing that our approach achieves much higher precision.

Now, we consider a more challenging adaptation task that needs various desired points (i.e., via-/end- points) in terms of quaternion and angular velocity. Note that orientation–DMP is not applicable in this case. As shown in Fig. 2(*c*), our approach indeed modulates quaternions and angular velocities to pass through various desired points. Specifically, the overall profiles of quaternion and angular velocity in demonstrations are maintained in the adaptation situation, which is a desirable feature of imitation learning.

B. Evaluations on Real Robot

We here consider a painting task that requires the real Barrett WAM robot to paint different areas with proper orientations. This task can be accomplished by concurrent adaptations of translational and rotational motions in Cartesian space, where the new painting area and its associated orientation can be guaranteed by new desired Cartesian positions and quaternions, in the form of via-point and end-point.

Through kinesthetic teaching (*first row* in Fig. 3), six

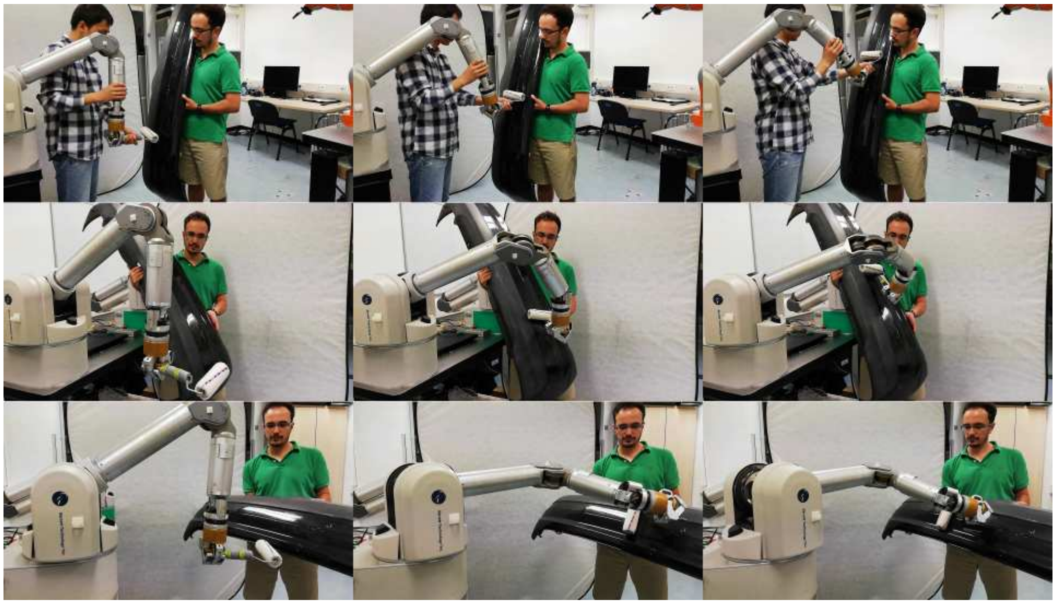


Fig. 3. Painting task on the real Barrett WAM robot. The *First row* shows the kinesthetic teaching of the painting task. The *Second row* and *third row* correspond to two adaptation evaluations, where new painting areas and orientations are unseen in demonstrations.

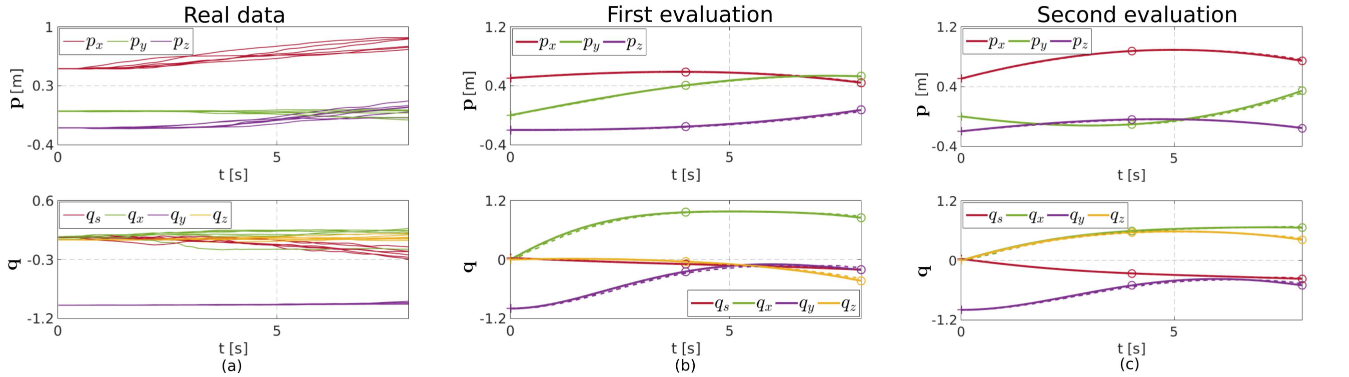


Fig. 4. Evaluations of our approach through a painting task on the real robot. (*a*) shows demonstrated Cartesian positions and quaternions in the painting task. (*b*)-(*c*) correspond to adapted Cartesian trajectories with various desired points, where solid curves represent planned trajectories by using our approach and dashed curves denote real measured trajectories. Circles depict desired Cartesian positions and quaternions.

demonstrations comprising time, Cartesian position and quaternion are recorded, as shown in Fig. 4(*a*). By following Algorithm 1, we can modulate demonstrated Cartesian trajectories towards various desired points[6]. We here consider two groups of evaluations and in each group two desired points (i.e., via-point and end-point) are defined. Note that all desired points in the evaluations are unseen in demonstrations. Besides, we set the initial Cartesian position and quaternion as an additional desired point so as to ensure smooth Cartesian trajectory at the very beginning of movement. The auxiliary quaternion $\mathbf{q}_a$ is set as the initial value of demonstrations. Other relevant parameters are $\ell = 0.001$ and $\lambda = 1$. The adapted Cartesian trajectories are shown in Fig. 4(*b*)-(*c*), where the planned trajectories (solid curves) and real measured trajectories (dashed curves) are provided. It can be seen that the planned trajectories are capable of meeting various constraints, i.e., Cartesian position and quaternion constraints. Snapshots of the painting task in new situations are shown in Fig. 3(*second and third rows*).

V. Conclusions

In this paper, we have proposed an approach for adapting quaternion and angular velocity towards arbitrary desired points, whose performance has been verified through several examples including simulated and real experiments. In comparison with previous works that mostly focus on orientation adaptation towards target points, our work allows for broader applications, particularly when both quaternion and angular velocity need to be modulated. Note that our method uses the kernel treatment instead of explicit basis functions (as used in [13], [14]), hence, it has potential in learning orientation profiles associated with high-dimensional inputs. We will focus on this topic in our future work.

Acknowledgement

We thank Dr. Bojan Nemec from Jožef Stefan Institute for his helpful feedback on the evaluation of orientation–DMP.

[6] For the case of adapting translational movement, we can treat $\mathbf{D}_\zeta$ and $\tilde{\mathbf{D}}_\zeta$ as demonstrated and desired Cartesian positions/velocities, respectively.

REFERENCES

[1] 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 International Conference on Intelligent Robots and Systems*, 2015, pp. 464-470.

[2] Y. Huang, B. Schölkopf and J. Peters, "Learning optimal striking points for a ping-pong playing robot," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 4587-4592.

[3] Y. Huang, D. Büchler, O. Koc, B. Schölkopf and J. Peters, "Jointly learning trajectory generation and hitting point prediction in robot table tennis," in *Proc. IEEE International Conference on Humanoid Robots*, 2016, pp. 650-655.

[4] S. Schaal, "Is imitation learning the route to humanoid robots," *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233-242, 1999

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

[6] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, "Probabilistic movement primitives," in *Proc. Advances in Neural Information Processing Systems*, 2013, pp. 2616-2624.

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

[8] Y. Huang, J. Silvério, L. Rozo, and D. G. Caldwell, "Generalized task-parameterized skill learning," in *Proc. International Conference on Robotics and Automation*, 2018, pp. 5667-5674.

[9] Y. Huang, L. Rozo, J. Silvério and D. G. Caldwell. "Kernelized movement primitives," *arXiv*:1708.08638v2, 2017.

[10] D. Koert, G. Maeda, R. Lioutikov, G. Neumann, and J. Peters, "Demonstration based trajectory optimization for generalizable robot motions," in *Proc. IEEE International Conference on Humanoid Robots*, 2016, pp. 515-522.

[11] Y. Zhou and T. Asfour, "Task-oriented generalization of dynamic movement primitive," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 3202-3209.

[12] P. Pastor, H. Hoffmann, T. Asfour and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *Proc. IEEE International Conference on Robotics and Automation*, 2009, pp. 763-768.

[13] A. Ude , B. Nemec , T. Petri and J. Morimoto, "Orientation in cartesian space dynamic movement primitives," in *Proc. IEEE International Conference on Robotics and Automation*, 2014, pp. 2997-3004.

[14] F. J. Abu-Dakka, B. Nemec, J. A. Jrgensen, T. R. Savarimuthu, N. Krger and A. Ude, "Adaptation of manipulation skills in physical contact with the environment to reference force profiles," *Autonomous Robots*, vol. 39, no. 2, pp. 199-217, 2015.

[15] H. Ravichandar and A. Dani, "Learning position and orientation dynamics from demonstrations via contraction analysis," *Autonomous Robots*, pp.1-16, 2018.

[16] S. Kim, R. Haschke and H. Ritter, "Gaussian mixture model for 3-dof orientations," *Robotics and Autonomous Systems*, vol. 87, pp. 28-37, 2017.

[17] M. J. Zeestraten, I. Havoutis, J. Silverio, S. Calinon, and D. G. Caldwell, "An approach for imitation learning on Riemannian manifolds," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1240-1247, 2017.

[18] M. Muhlig, M. Gienger, S. Hellbach, J. J. Steil and C. Goerick, "Task-level imitation learning using variance-based movement optimization," in *Proc. IEEE International Conference on Robotics and Automation*, 2009, pp. 1177-1184.

[19] E. Todorov and M. I. Jordan, "Optimal feedback control as a theory of motor coordination," *Nature Neuroscience*, vol. 5, no. 11, p.1226-1235, 2002.

[20] J. R. Medina, D. Lee and S. Hirche, "Risk-sensitive optimal feedback control for haptic assistance," in *Proc. IEEE International Conference on Robotics and Automation*, 2012, pp. 1025-1031.

[21] Y. Huang, J. Silvério, S. Calinon, "Towards Minimal Intervention Control with Competing Constraints," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 733-738.

[22] D. A. Cohn, Z. Ghahramani and M. I. Jordan Active learning with statistical models. *Journal of Artifial Intelligence Research*, vol. 4, no. 1, pp. 129-145, 1996.

[23] Y. Huang, L. Rozo, J. Silvério and D. G. Caldwell. "Non-parametric imitation learning of robot motor skills," in *Proc. IEEE International Conference on Robotics and Automation*, 2019.

[24] C. Saunders, A. Gammerman, V. Vovk, "Ridge Regression Learning Algorithm in Dual Variables," in *Proc. International Conference on Machine Learning*, 1998, pp. 515-521.

[25] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*, Chapter 14.4.3, pp. 492-493, The MIT Press, 2012.

[26] J. Kober, E. Öztop and J. Peters, "Reinforcement learning to adjust robot movements to new situations," in *Proc. International Joint Conference on Artificial Intelligence*, 2011, pp. 2650-2655.