Bimanual Skill Learning with Pose and Joint Space Constraints

João Silvério[1], Sylvain Calinon[2,1], Leonel Rozo[1] and Darwin G. Caldwell[1]

***Abstract*— As humanoid robots become commonplace, learning and control algorithms must take into account the new challenges imposed by this morphology, in order to fully exploit their potential. One of the most prominent characteristics of such robots is their bimanual structure. Most research on learning bimanual skills has focused on the coordination between end-effectors, exploiting operational space formulations. However, motion patterns in bimanual scenarios are not exclusive to operational space, also occurring at joint level. Moreover, in addition to position, the end-effector orientation is also essential for bimanual operation. Here, we propose a framework for simultaneously learning constraints in configuration and operational spaces, while considering end-effector orientations, commonly overlooked in previous works. In particular, we extend the Task-Parameterized Gaussian Mixture Model (TP-GMM) with novel Jacobian-based operators that address the foregoing problem. The proposed framework is evaluated in a bimanual task with the COMAN humanoid that requires the consideration of operational and configuration space motions.**

I. INTRODUCTION

The human-robot transfer of bimanual skills is a growing topic of research in robot learning. As the number of available dual-arm platforms and humanoid robots increases, existing learning and control techniques must be adapted to take full advantage of the repertoire of tasks that such robots can perform [1]. Among the available learning frameworks, Programming by Demonstration (PbD) [2] is a promising direction, but historically it has mostly addressed single-arm skills. Recent works attempt to bridge the gap and focus on PbD for bimanual manipulation, exploiting operational space formulations (e.g. [3], [4], [5], [6], [7], [8]), with a strong focus on extracting arm coordination and/or dominance. However, when skills contain constraints in configuration space, such as preferred postures or arm movements for which joint trajectories are more important, operational space formulations alone are insufficient for correct task execution. Classical control approaches (see e.g. [9]) exploit the null space of tasks in Cartesian space to perform secondary motions in joint space. Such approaches enforce strict priority hierarchies, which do not allow the joint space to have the highest priority, and will only perform adequately while the system is redundant with respect to the main task. Figure 1 shows an example where the two types of constraints are present. The depicted skill consists of a phase where the operational space dominates over configuration space (approaching and grasping the bottle) and another phase where the configuration space takes over (shaking through rhythmic movements of the shoulder joints). Our approach is aimed at allowing the robot to learn the importance of the two types of constraints from the demonstrations, regardless of redundancy or previously set hierarchies.

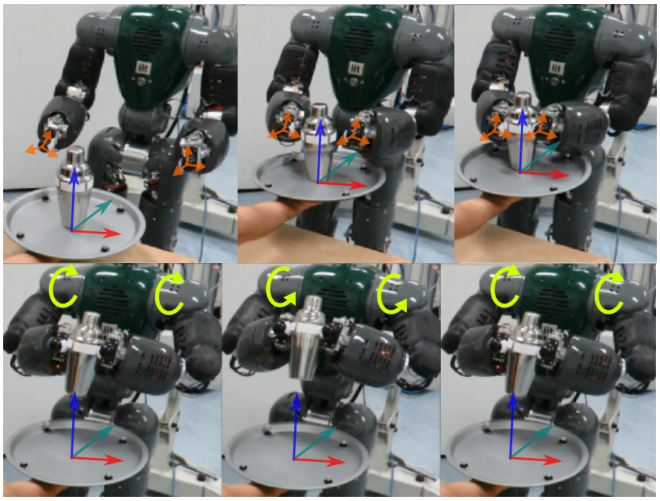

Fig. 1: The COMAN robot performs a bimanual shaking task. **Top:** snapshots of the reaching part of the movement, governed by constraints in operational space. The robot gradually reaches for the object, which is tracked by an optical system. **Bottom:** shaking part, defined by constraints in configuration space. The robot performs the shaking through rhythmic motions of the shoulders, moving the shaker up and down.

Endowing robots with the ability to learn how to handle possibly competing constraints in operational and configuration spaces is thus an important challenge in the learning of robot controllers. Earlier work from Calinon [10] addresses this problem in single-arm tasks. One limitation of [10] is that the approach only accounts for position constraints in operational space. Hence, by design, it prevents orientation motions with respect to objects (or between hands) from being learned by the robot. The role of orientation in bimanual manipulation has been actively studied in recent years [7], [8]. Indeed, it is widely accepted that being able to transfer orientation abilities to robots performing bimanual skills is crucial for proper execution of most daily life tasks.

This paper extends [10] to consider orientation when learning configuration and operational space constraints. We draw insights from [8] and take advantage of quaternion algebra to project operational space motions onto configuration space, allowing for the extraction of consistent orientation patterns from demonstrations. On the basis of [8], the problem is framed as a Task-Parameterized Gaussian Mixture Model (TP-GMM) [11], reviewed in Section II, whose structure is exploited to consider Jacobian matrices and quaternion-based

[1]Department of Advanced Robotics, Istituto Italiano di Tecnologia, 16163 Genova, Italy (e-mail: name.surname@iit.it).

[2]Idiap Research Institute, CH-1920 Martigny, Switzerland (e-mail: sylvain.calinon@idiap.ch).

The authors would like to thank Luca Muratore and Phil Hudson for their assistance in the experiments with the COMAN robot.

operators, resulting in a principled and self-contained solution to the foregoing problem (Section III). We validate the proposed approach on the COMpliant huMANoid (COMAN) [12]. For this, we choose the skill of bimanually shaking a bottle (Figure 1). The experimental results are reported in Section IV, which is followed by concluding remarks in Section V.

II. TASK-PARAMETERIZED GAUSSIAN MIXTURE MODELS (TP-GMM)

In previous work [11] we introduced a probabilistic approach to task-parameterized movements, the Task-Parameterized Gaussian Mixture Model (TP-GMM). TP-GMM was used to encode demonstrated end-effector motions in multiple coordinate systems simultaneously, whose importance changed during the task depending on the variability observed in the demonstrations. This information was exploited to adapt demonstrated skills to new situations. In this section we review TP-GMM.

A. Task parameters

Definition 1: Task parameters are sets of linear operators $\boldsymbol{A}^{(j)}, \boldsymbol{b}^{(j)}$ that map Gaussian distributions from P subspaces, indexed by $j = 1, \dots, P$, onto a common space. Such sets are called *'task parameters'* since they are part of the parameterization of a task, i.e. they influence how the robot accomplishes the given task goals.

In TP-GMM, P subspaces encode local features of a demonstrated skill. Section II-B describes how local models of features are trained. Once projected onto a common space through the *task parameters*, the local models are combined to yield a solution that fulfills the most important features at every moment of task execution (Section II-C).

In previous work, task parameters have been used to represent poses of objects in a robot workspace, mapping local models of demonstrations (from the perspective of P different objects) onto a global frame—the robot base frame. In these cases, $\boldsymbol{A}^{(j)}$ is a rotation matrix [3], [11] or a quaternion matrix [8], and $\boldsymbol{b}^{(j)}$ is a translation vector, representing respectively the object orientation and origin of its coordinate system with respect to the robot base frame. Note that, in previous work, authors referred to task parameters as *candidate frames* or *candidate coordinate systems*. This is because, for a given task, each set of task parameters $\boldsymbol{A}^{(j)}, \boldsymbol{b}^{(j)}$ may or may not influence the task execution, depending on the variability of the teacher's demonstrations in the corresponding coordinate system. Hence, each set $\boldsymbol{A}^{(j)}, \boldsymbol{b}^{(j)}$ is considered to be a *candidate* for affecting the task outcome.

In this work, we exploit the linear structure inherent to TP-GMM, through the task parameters, to address the problem of combining constraints between configuration and operational spaces, with orientation. We do this by taking a different perspective from past work—where task parameters represented coordinate systems—and devising formulations of task parameters that correspond to *candidate projection operators* (Section III).

B. Model estimation

Formally, each demonstration $m \in \{1, \dots, M\}$ contains T_m datapoints of dimension D forming a dataset of N datapoints $\{\boldsymbol{\xi}_t\}_{t=1}^{N}$ with $N = \sum_{m=1}^{M} T_m$ and $\boldsymbol{\xi}_t \in \mathbb{R}^D$. P task parameters, that map between subspaces $j = 1, \dots, P$ and a common space, are defined at every time step t by $\{\boldsymbol{A}_t^{(j)}, \boldsymbol{b}_t^{(j)}\}_{j=1}^{P}$. The demonstrations $\boldsymbol{\xi} \in \mathbb{R}^{D \times N}$ are observed from each subspace, forming P local datasets $\boldsymbol{X}^{(j)} \in \mathbb{R}^{D \times N}$. As an example, in previous work [3], [11], $\boldsymbol{\xi}_t$ corresponded to end-effectors positions, and the task parameters represented coordinate systems, thus the local datasets were computed from $\boldsymbol{X}_t^{(j)} = \boldsymbol{A}_t^{(j)^{-1}} \left(\boldsymbol{\xi}_t - \boldsymbol{b}_t^{(j)}\right)$ since $\{\boldsymbol{A}_t^{(j)}, \boldsymbol{b}_t^{(j)}\}_{j=1}^{P}$ parameterized the orientations and positions of P objects.

The *model parameters* of a TP-GMM with K components are defined by $\left\{\pi_i, \{\boldsymbol{\mu}_i^{(j)}, \boldsymbol{\Sigma}_i^{(j)}\}_{j=1}^{P}\right\}_{i=1}^{K}$, where π_i are the mixing coefficients and $\boldsymbol{\mu}_i^{(j)}$, $\boldsymbol{\Sigma}_i^{(j)}$ denote the center and covariance matrix of the i-th Gaussian from subspace j. Learning of the model parameters is achieved by *expectation-maximization* (EM), see [11] for details.

C. Gaussian Mixture Regression

The learned model is used to reproduce movements in new situations. Each subspace encodes local features of the demonstrated movement. In new situations, i.e. new values of the task parameters $\boldsymbol{A}_t^{(j)}, \boldsymbol{b}_t^{(j)}$, one needs to find a trade-off between each local solution. TP-GMM solves this problem by combining the local models, projected in the common space, using the product of Gaussians. In this way, a new GMM with parameters $\{\pi_i, \hat{\boldsymbol{\mu}}_{t,i}, \hat{\boldsymbol{\Sigma}}_{t,i}\}_{i=1}^{K}$ is automatically generated as

$$\mathcal{N}\Big(\hat{\boldsymbol{\mu}}_{t,i}, \hat{\boldsymbol{\Sigma}}_{t,i}\Big) \propto \prod_{j=1}^{P} \mathcal{N}\Big(\hat{\boldsymbol{\mu}}_{t,i}^{(j)}, \hat{\boldsymbol{\Sigma}}_{t,i}^{(j)}\Big), \text{ with}$$
$$\hat{\boldsymbol{\mu}}_{t,i}^{(j)} = \boldsymbol{A}_t^{(j)} \boldsymbol{\mu}_i^{(j)} + \boldsymbol{b}_t^{(j)}, \quad \hat{\boldsymbol{\Sigma}}_{t,i}^{(j)} = \boldsymbol{A}_t^{(j)} \boldsymbol{\Sigma}_i^{(j)} \boldsymbol{A}_t^{(j)^{\mathsf{T}}}, \quad (1)$$

where the result of the Gaussian product is given by

$$\hat{\boldsymbol{\Sigma}}_{t,i} = \Big(\sum_{j=1}^{P} \hat{\boldsymbol{\Sigma}}_{t,i}^{(j)^{-1}}\Big)^{-1}, \quad \hat{\boldsymbol{\mu}}_{t,i} = \hat{\boldsymbol{\Sigma}}_{t,i} \sum_{j=1}^{P} \hat{\boldsymbol{\Sigma}}_{t,i}^{(j)^{-1}} \hat{\boldsymbol{\mu}}_{t,i}^{(j)}. \quad (2)$$

Equation (1) maps local models onto a common space space, where information from the different subspaces is fused by computing a product of Gaussians. Note that task parameters $\boldsymbol{A}_t^{(j)}, \boldsymbol{b}_t^{(j)}$ may vary during reproduction and take values different from those observed during demonstrations. In previous work [8], [3], this property was exploited to adapt demonstrated skills to new situations (e.g., unobserved positions and orientations of manipulated objects). The obtained GMM is used to retrieve a reference for the robot through Gaussian Mixture Regression (GMR) at any given time step t. In this case, the datapoint $\boldsymbol{\xi}_t$ is decomposed into two subvectors $\boldsymbol{\xi}_t^{\mathcal{I}}$ and $\boldsymbol{\xi}_t^{\mathcal{O}}$, respectively, spanning the input and output dimensions of the regression problem, thus the new GMM encodes the joint probability distribution

$\mathcal{P}(\boldsymbol{\xi}_t^{\mathcal{I}}, \boldsymbol{\xi}_t^{\mathcal{O}}) \sim \sum_{i=1}^{K} \pi_i \, \mathcal{N}\big(\hat{\boldsymbol{\mu}}_{t,i}, \hat{\boldsymbol{\Sigma}}_{t,i}\big)$. GMR thus generates a new distribution $\mathcal{P}(\boldsymbol{\xi}_t^{\mathcal{O}} | \boldsymbol{\xi}_t^{\mathcal{I}}) = \mathcal{N}\big(\boldsymbol{\xi}_t^{\mathcal{O}} | \hat{\boldsymbol{\mu}}_t^{\mathcal{O}}, \hat{\boldsymbol{\Sigma}}_t^{\mathcal{O}}\big)$ that is used to control the robot. A more comprehensive description of GMR in the context of TP-GMM, can be found in [11].

In the next section we show how Jacobian matrices, commonly employed in robotics as linear mapping operators in differential kinematics, can be exploited as task parameters.

III. LEARNING OPERATIONAL AND CONFIGURATION SPACE CONSTRAINTS

Previous works that exploited TP-GMM [8], [3], [11] considered manipulation problems defined in operational space. In such cases, task parameters were related to object poses, i.e., $\boldsymbol{b}_t^{(j)}$ and $\boldsymbol{A}_t^{(j)}$ parameterized positions and orientations of coordinate systems. In this section, we propose to exploit the structure of TP-GMM described in Section II to simultaneously consider constraints in operational and configuration spaces. We do this by introducing Jacobian-based task parameters (formulated in Sections III-A and III-B for position and orientation, respectively) that project operational space constraints on configuration space (Section III-C), where Gaussian products (1), (2) are computed.

A. Jacobian-based task parameters for position constraints

Handling constraints in configuration and operational spaces is achieved by exploiting the linear structure of the task parameters of TP-GMM, in combination with inverse kinematics. Formally, consider a manipulator with N_q joints, whose positions and velocities are denoted by $\boldsymbol{q}, \dot{\boldsymbol{q}} \in \mathbb{R}^{N_q}$. Its differential kinematics are given by $\begin{bmatrix}\dot{\boldsymbol{x}}^{\mathsf{T}} & \boldsymbol{\omega}^{\mathsf{T}}\end{bmatrix}^{\mathsf{T}} = \boldsymbol{J}\dot{\boldsymbol{q}}$, where $\dot{\boldsymbol{x}}, \boldsymbol{\omega} \in \mathbb{R}^3$ are the operational space linear and angular velocities, respectively. The Jacobian matrix $\boldsymbol{J} = \begin{bmatrix}\boldsymbol{J}_p^{\mathsf{T}} & \boldsymbol{J}_o^{\mathsf{T}}\end{bmatrix}^{\mathsf{T}} \in \mathbb{R}^{6 \times N_q}$ accounts for the contribution of joint velocities to operational space velocities, with matrices $\boldsymbol{J}_p, \boldsymbol{J}_o \in \mathbb{R}^{3 \times N_q}$ responsible for the linear and angular parts, respectively. The inverse differential kinematics equation $\dot{\boldsymbol{q}} = \boldsymbol{J}^{\dagger}\dot{\boldsymbol{x}}$, with $\boldsymbol{J}^{\dagger} = \boldsymbol{J}^{\mathsf{T}}\left(\boldsymbol{J}\boldsymbol{J}^{\mathsf{T}}\right)^{-1}$ the right pseudo-inverse of the manipulator Jacobian $\boldsymbol{J}$, yields the minimum-norm $\dot{\boldsymbol{q}}$ that ensures $\dot{\boldsymbol{x}}$ in operational space [13].

The inverse differential kinematics for the position part is given by $\dot{\boldsymbol{q}} = \boldsymbol{J}_p^{\dagger}\dot{\boldsymbol{x}}$. Numerical integration of this equation permits the computation of joint references for a desired end-effector position $\boldsymbol{x}_t$ as (dropping the Jacobian subscript p)

$$\begin{aligned} &\hat{\boldsymbol{q}}_t - \boldsymbol{q}_{t-1} = \boldsymbol{J}_{t-1}^{\dagger}\left(\boldsymbol{x}_t - \boldsymbol{x}_{t-1}\right) \\ &\iff \hat{\boldsymbol{q}}_t = \boldsymbol{J}_{t-1}^{\dagger}\boldsymbol{x}_t + \boldsymbol{q}_{t-1} - \boldsymbol{J}_{t-1}^{\dagger}\boldsymbol{x}_{t-1}, \end{aligned} \tag{3}$$

where $\hat{\boldsymbol{q}}_t$ denotes the desired joint angles at t. The structure of (3), being affine in $\boldsymbol{x}_t$, allows us to connect inverse kinematics with TP-GMM. Let us assume, for the sake of the argument, that end-effector positions are modeled as trajectory distributions $\boldsymbol{x}_t \sim \sum_{i=1}^{K} \pi_i \mathcal{N}\left(\boldsymbol{\mu}_i^{(j)}, \boldsymbol{\Sigma}_i^{(j)}\right)$. The index j denotes one arbitrary subspace, as discussed in Section II, where robot end-effector positions are locally modeled by a GMM. It follows that the linear transformation properties of Gaussian distributions (1) can be applied to (3) to project the local GMM on configuration space, resulting in

$$\hat{\boldsymbol{q}}_{t,i}^{(j)} = \underbrace{\boldsymbol{J}_{t-1}^{\dagger}}_{\boldsymbol{A}_t^{(j)}}\boldsymbol{\mu}_i^{(j)} + \underbrace{\boldsymbol{q}_{t-1} - \boldsymbol{J}_{t-1}^{\dagger}\boldsymbol{x}_{t-1}}_{\boldsymbol{b}_t^{(j)}}, \quad \forall i = 1, \ldots, K, \tag{4}$$

for the mean of state i, and

$$\hat{\boldsymbol{\Sigma}}_{t,i}^{(j)} = \underbrace{\boldsymbol{J}_{t-1}^{\dagger}}_{\boldsymbol{A}_t^{(j)}}\boldsymbol{\Sigma}_i^{(j)}\underbrace{\left(\boldsymbol{J}_{t-1}^{\dagger}\right)^{\mathsf{T}}}_{\boldsymbol{A}_t^{(j)\mathsf{T}}}, \tag{5}$$

for the corresponding covariance matrix. Equations (4) and (5) show that Jacobian-based task parameters $\boldsymbol{A}_t^{(j)}, \boldsymbol{b}_t^{(j)}$ map Gaussian distributions from operational to configuration space, creating new distributions of joint angles. The linear structure of TP-GMM is, hence, not exclusive to the use of coordinate system representations of task parameters.

This result is the cornerstone of more complex types of operational space projection operators. For instance, if we consider end-effector positions encoded with respect to an object parameterized by a translation vector $\boldsymbol{p}_t^{(j)}$ and a rotation matrix $\boldsymbol{R}_t^{(j)}$, (4) becomes

$$\hat{\boldsymbol{q}}_{t,i}^{(j)} = \underbrace{\boldsymbol{J}_{t-1}^{\dagger}\boldsymbol{R}_t^{(j)}}_{\boldsymbol{A}_t^{(j)}}\boldsymbol{\mu}_i^{(j)} + \underbrace{\boldsymbol{J}_{t-1}^{\dagger}\left(\boldsymbol{p}_t^{(j)} - \boldsymbol{x}_{t-1}\right) + \boldsymbol{q}_{t-1}}_{\boldsymbol{b}_t^{(j)}}, \tag{6}$$

which can be derived in a straightforward manner from (4) by assuming rotated and translated Gaussians with mean $\boldsymbol{R}_t^{(j)}\boldsymbol{\mu}_i^{(j)} + \boldsymbol{p}_t^{(j)}$ and covariance $\boldsymbol{R}_t^{(j)}\boldsymbol{\Sigma}_i^{(j)}\boldsymbol{R}_t^{(j)^{\mathsf{T}}}$. Similarly, the expression for the covariance matrix $\hat{\boldsymbol{\Sigma}}_{t,i}^{(j)}$ can be easily obtained based on (5), employing $\boldsymbol{A}_t^{(j)}$ as defined in (6).

On the basis of this construction of task parameters, the local datasets $\boldsymbol{X}^{(j)} = [\boldsymbol{X}_1^{(j)}, \ldots, \boldsymbol{X}_N^{(j)}]$ used to train the TP-GMM model can be computed in a similar fashion to Section II, for standard coordinate systems. If a subspace j models the absolute end-effector position, i.e. with respect to the robot base frame, we have $\boldsymbol{X}_t^{(j)} = \boldsymbol{x}_t$. On the other hand, when j is associated with a coordinate system with pose parameters $\boldsymbol{p}_t^{(j)}, \boldsymbol{R}_t^{(j)}$ as in (6), we have $\boldsymbol{X}_t^{(j)} = \boldsymbol{R}_t^{(j)^{\mathsf{T}}}\left(\boldsymbol{x}_t - \boldsymbol{p}_t^{(j)}\right)$.

In summary, in this novel formulation of task parameters, $\boldsymbol{A}_t^{(j)} \in \mathbb{R}^{N_q \times 3}$ and $\boldsymbol{b}_t^{(j)} \in \mathbb{R}^{N_q}$ map from Cartesian position to joint angles, solving the inverse kinematics with a reference given by the mean $\boldsymbol{\mu}_i^{(j)}$. The TP-GMM representation is therefore extended to Jacobian-based task parameters $\{\boldsymbol{b}_t^{(j)}, \boldsymbol{A}_t^{(j)}\}_{j=1}^{P}$ with non-square $\boldsymbol{A}_t^{(j)}$ matrices. With this representation, operational space constraints are projected on configuration space, where Gaussian products can be computed as in (2), extending the original TP-GMM formulation to the consideration of configuration space constraints.

B. Orientation projection operators

The previous subsection showed how we can formulate task parameters to project end-effector position constraints

TABLE I: Summary of task parameters as candidate projection operators

Configuration space constraints:	$\hat{\boldsymbol{q}}_{t,i}^{(j)} = \boldsymbol{I}$	$\boldsymbol{\mu}_i^{(j)} + \mathbf{0}$
Absolute position constraints:	$\hat{\boldsymbol{q}}_{t,i}^{(j)} = \boldsymbol{J}_{p,t-1}^{\dagger}$	$\boldsymbol{\mu}_i^{(j)} - \boldsymbol{J}_{p,t-1}^{\dagger}\boldsymbol{x}_{t-1} + \boldsymbol{q}_{t-1}$
Relative position constraints:	$\hat{\boldsymbol{q}}_{t,i}^{(j)} = \boldsymbol{J}_{p,t-1}^{\dagger}\boldsymbol{R}_t^{(j)}$	$\boldsymbol{\mu}_i^{(j)} + \boldsymbol{J}_{p,t-1}^{\dagger}(\boldsymbol{p}_t^{(j)} - \boldsymbol{x}_{t-1}) + \boldsymbol{q}_{t-1}$
Absolute orientation constraints:	$\hat{\boldsymbol{q}}_{t,i}^{(j)} = \boldsymbol{J}_{o,t-1}^{\dagger}\bar{\boldsymbol{H}}^{*}(\bar{\boldsymbol{\epsilon}}_{t-1})$	$\boldsymbol{\mu}_i^{(j)} + \boldsymbol{q}_{t-1}$
Relative orientation constraints:	$\hat{\boldsymbol{q}}_{t,i}^{(j)} = \underbrace{\boldsymbol{J}_{o,t-1}^{\dagger}\bar{\boldsymbol{H}}^{*}(\bar{\boldsymbol{\epsilon}}_{t-1})\overset{+}{\boldsymbol{H}}(\boldsymbol{\epsilon}_t^{(j)})}_{\boldsymbol{A}_t^{(j)}}$	$\boldsymbol{\mu}_i^{(j)} + \underbrace{\boldsymbol{q}_{t-1}}_{\boldsymbol{b}_t^{(j)}}$

on configuration space. Here, we take advantage of the algebraic properties of unit quaternions to derive linear operators for projecting orientation constraints. Let us consider the orientation part of the end-effector pose represented by a unit quaternion $\boldsymbol{\epsilon}$ (the Appendix reviews this representation) and the inverse differential kinematics for angular velocities, $\dot{\boldsymbol{q}} = \boldsymbol{J}_o^{\dagger}\boldsymbol{\omega}$. From [14] we have that

$$\boldsymbol{\omega}_t \approx \frac{\mathrm{vec}(\boldsymbol{\epsilon}_t * \bar{\boldsymbol{\epsilon}}_{t-1})}{\Delta t} \tag{7}$$

gives the angular velocity that rotates the unit quaternion $\boldsymbol{\epsilon}_{t-1}$ into $\boldsymbol{\epsilon}_t$, during Δt. Note that the non-linearity of the operator $\mathrm{vec}(\boldsymbol{\epsilon}_t * \bar{\boldsymbol{\epsilon}}_{t-1})$ is incompatible with the structure of TP-GMM parameterization. We can however employ unit quaternion properties (see Appendix) to simplify this operator. For any unit quaternion, $\mathrm{vec}(\boldsymbol{\epsilon})$ can be replaced by the matrix operation $\mathrm{vec}(\boldsymbol{\epsilon}) = \begin{bmatrix}\mathbf{0}^{3\times 1} & \boldsymbol{I}^{3\times 3}\end{bmatrix}\boldsymbol{\epsilon}$, allowing us to rewrite (7) as

$$\boldsymbol{\omega}_t = \begin{bmatrix}\mathbf{0}^{3\times 1} & \boldsymbol{I}^{3\times 3}\end{bmatrix}(\boldsymbol{\epsilon}_t * \bar{\boldsymbol{\epsilon}}_{t-1})\frac{1}{\Delta t}. \tag{8}$$

The quaternion product $\boldsymbol{\epsilon}_t * \bar{\boldsymbol{\epsilon}}_{t-1}$ can also be replaced by a matrix product using quaternion matrices. For this, we take advantage of the operator $\bar{\boldsymbol{H}}$ (20) that allows for changing the order in which two quaternions are multiplied without changing the resulting orientation. Hence, we have

$$\boldsymbol{\epsilon}_t * \bar{\boldsymbol{\epsilon}}_{t-1} = \bar{\boldsymbol{H}}(\bar{\boldsymbol{\epsilon}}_{t-1})\,\boldsymbol{\epsilon}_t. \tag{9}$$

Replacing in (8) yields

$$\boldsymbol{\omega}_t = \begin{bmatrix}\mathbf{0}^{3\times 1} & \boldsymbol{I}^{3\times 3}\end{bmatrix}\bar{\boldsymbol{H}}(\bar{\boldsymbol{\epsilon}}_{t-1})\,\boldsymbol{\epsilon}_t\frac{1}{\Delta t} \tag{10}$$

$$= \bar{\boldsymbol{H}}^{*}(\bar{\boldsymbol{\epsilon}}_{t-1})\,\boldsymbol{\epsilon}_t\frac{1}{\Delta t}\,, \tag{11}$$

where $\bar{\boldsymbol{H}}^{*}(\bar{\boldsymbol{\epsilon}}_{t-1}) = \begin{bmatrix}\mathbf{0}^{3\times 1} & \boldsymbol{I}^{3\times 3}\end{bmatrix}\bar{\boldsymbol{H}}(\bar{\boldsymbol{\epsilon}}_{t-1})$, allowing us to write the inverse kinematics equation as (dropping the subscript o in the Jacobian matrix)

$$\hat{\dot{\boldsymbol{q}}}_t = \boldsymbol{J}_{t-1}^{\dagger}\bar{\boldsymbol{H}}^{*}(\bar{\boldsymbol{\epsilon}}_{t-1})\,\boldsymbol{\epsilon}_t\frac{1}{\Delta t}\,, \tag{12}$$

$$\iff \hat{\boldsymbol{q}}_t = \boldsymbol{J}_{t-1}^{\dagger}\bar{\boldsymbol{H}}^{*}(\bar{\boldsymbol{\epsilon}}_{t-1})\,\boldsymbol{\epsilon}_t + \boldsymbol{q}_{t-1}, \tag{13}$$

which has a similar structure[‡] to (3), being linear on the quaternion $\boldsymbol{\epsilon}_t$. In a similar way to Section III-A, if $\boldsymbol{\mu}_i^{(j)}$ is the mean of a Gaussian i, encoding the absolute orientation of the end-effector in a subspace j, we take advantage of the structure in (13) to devise new task parameters $\boldsymbol{A}_t^{(j)}, \boldsymbol{b}_t^{(j)}$ that map a GMM from quaternion space to configuration space, namely

$$\hat{\boldsymbol{q}}_{t,i}^{(j)} = \underbrace{\boldsymbol{J}_{t-1}^{\dagger}\bar{\boldsymbol{H}}^{*}(\bar{\boldsymbol{\epsilon}}_{t-1})}_{\boldsymbol{A}_t^{(j)}}\,\boldsymbol{\mu}_i^{(j)} + \underbrace{\boldsymbol{q}_{t-1}}_{\boldsymbol{b}_t^{(j)}}, \tag{14}$$

for the mean of state i (the covariance can be derived by following the same rules explained in Section III-A).

For a desired end-effector orientation encoded in a coordinate system whose orientation is given by $\boldsymbol{\epsilon}_t^{(j)}$, (14) becomes

$$\hat{\boldsymbol{q}}_{t,i}^{(j)} = \underbrace{\boldsymbol{J}_{t-1}^{\dagger}\bar{\boldsymbol{H}}^{*}\left(\bar{\boldsymbol{\epsilon}}_{t-1}\right)\overset{+}{\boldsymbol{H}}\left(\boldsymbol{\epsilon}_t^{(j)}\right)}_{\boldsymbol{A}_t^{(j)}}\boldsymbol{\mu}_i^{(j)} + \underbrace{\boldsymbol{q}_{t-1}}_{\boldsymbol{b}_t^{(j)}}, \tag{15}$$

where $\overset{+}{\boldsymbol{H}}$ is a quaternion matrix (see Appendix A).

For this formulation, if a subspace j models the absolute end-effector orientation, i.e. with respect to the base frame of the robot, we have $\boldsymbol{X}_t^{(j)} = \boldsymbol{\epsilon}_t$. When j is associated with a coordinate system with quaternion $\boldsymbol{\epsilon}_t^{(j)}$, then $\boldsymbol{X}_t^{(j)} = \overset{+}{\boldsymbol{H}}\left(\bar{\boldsymbol{\epsilon}}_t^{(j)}\right)\boldsymbol{\epsilon}_t$.

C. Task parameters for configuration space constraints

The previous two subsections provided task parameters that project operational space constraints on configuration space. However, in order to consider both operational and configuration space constraints, one requires a local model of the configuration space demonstrations as well. Encoding configuration space movements in a TP-GMM is done using simple task parameters $\boldsymbol{A}_t^{(j)} = \boldsymbol{I},\ \boldsymbol{b}_t^{(j)} = \mathbf{0}$, corresponding to a canonical projection operator. This stems from the fact that, in this case, the subspace is the configuration space itself, i.e. , $\hat{\boldsymbol{q}}_{t,i}^{(j)} = \boldsymbol{\mu}_i^{(j)}$. The local datasets are thus computed from $\boldsymbol{X}_t^{(j)} = \boldsymbol{q}_t$, where $\boldsymbol{q}_t \in \mathbb{R}^{N_q}$ is the vector of robot joint angles at time step t.

Table I gives a summary of the operators derived in this section. The overall procedure for learning and reproducing a skill is summarized in Algorithm 1.

IV. EXPERIMENTAL RESULTS

In order to test the operators introduced in Section III, we selected the bimanual skill of shaking a bottle using COMAN. The whole skill contains an operational space component (reaching, grasping a bottle and bringing it closer

[‡]Note that the logarithmic map of the unit quaternion (see [14]), could alternatively be used instead of (7). However, the linear structure of Eq. (11) makes (7) a convenient form for the TP-GMM parameterization.

Algorithm 1 Simultaneously learning constraints in operational and configuration spaces

Initialization

1: Select candidate projection operators from Table I based on the task at hand
- Canonical operator $\boldsymbol{A}_t^{(j)} = \boldsymbol{I}$, $\boldsymbol{b}_t^{(j)} = \boldsymbol{0}$, for encoding configuration space constraints
- Operational space operators, for absolute or relative position/orientation constraints in operational space

2: Collect demonstrations and compute the local datasets $\boldsymbol{X}^{(j)}$ according to the chosen operators

Model training

1: Apply EM [11] to obtain $\left\{\pi_i, \{\boldsymbol{\mu}_i^{(j)}, \boldsymbol{\Sigma}_i^{(j)}\}_{j=1}^{P}\right\}_{i=1}^{K}$

Movement synthesis

1: **for** $t = 1, \ldots, N$ **do**
2: **for** $j = 1, \ldots, P$ **do**
3: Update $\{\boldsymbol{A}_t^{(j)}, \boldsymbol{b}_t^{(j)}\}$ according to Table I
4: **end for**
5: **for** $i = 1, \ldots, K$ **do**
6: Compute $\hat{\boldsymbol{\mu}}_{t,i}$ and $\hat{\boldsymbol{\Sigma}}_{t,i}$ from (1) and (2)
7: **end for**
8: Apply GMR at $\boldsymbol{\xi}_t^{\mathcal{I}}$: $\mathcal{P}(\boldsymbol{\xi}_t^{\mathcal{O}} | \boldsymbol{\xi}_t^{\mathcal{I}}) = \mathcal{N}\left(\boldsymbol{\xi}_t^{\mathcal{O}} | \hat{\boldsymbol{\mu}}_t^{\mathcal{O}}, \hat{\boldsymbol{\Sigma}}_t^{\mathcal{O}}\right)$
9: Use $\hat{\boldsymbol{\mu}}_t^{\mathcal{O}}$ as joint references for the robot controller
10: **end for**

to the torso) and a configuration space component (shaking with rhythmic shoulder movements). The presented results were obtained in the Gazebo simulator, however, the skill was also reproduced in the real robot (Figure 1). The used demonstrations were generated in Gazebo (Figure 2) solving inverse kinematics, for the operational space part of the movement, and using sinusoidal references to control the shoulder joints, for the shaking part*. Here, we assume that the demonstrated grasp is always successful and the object will move together with the end-effectors after it is grasped. Hence, the pose of the bottle that is considered in this experiment is the one at the beginning of each demonstration. Moreover, we also assume that the grasping points on the bottle are the same in all demonstrations. Videos of the reproduction in the real robot are available at `http://joaosilverio.weebly.com/Humanoids18` .

A. Setup

The upper-body of the COMAN robot comprises 17 DOFs: 3 DOFs for the waist and 7 for each arm, with the kinematic chains of both arms sharing the 3 waist joints. We define the differential kinematics of the left and right end-effectors as $\begin{bmatrix}\dot{\boldsymbol{x}}_L^\mathsf{T} & \boldsymbol{\omega}_L^\mathsf{T} & \dot{\boldsymbol{x}}_R^\mathsf{T} & \boldsymbol{\omega}_R^\mathsf{T}\end{bmatrix}^\mathsf{T} = \boldsymbol{J}_{\text{up}}\dot{\boldsymbol{q}}$, where $\boldsymbol{J}_{\text{up}}$ is the upper-body Jacobian, $\dot{\boldsymbol{x}}_L, \boldsymbol{\omega}_L, \dot{\boldsymbol{x}}_R, \boldsymbol{\omega}_R$ are the left and right end-effector velocities and $\dot{\boldsymbol{q}} = \begin{bmatrix}\dot{\boldsymbol{q}}_W^\mathsf{T} & \dot{\boldsymbol{q}}_L^\mathsf{T} & \dot{\boldsymbol{q}}_R^\mathsf{T}\end{bmatrix}^\mathsf{T}$ represents the concatenation of waist, left and right arm joint velocities.

*Alternatively, kinesthetic teaching or optical tracking of movements from humans could be used.

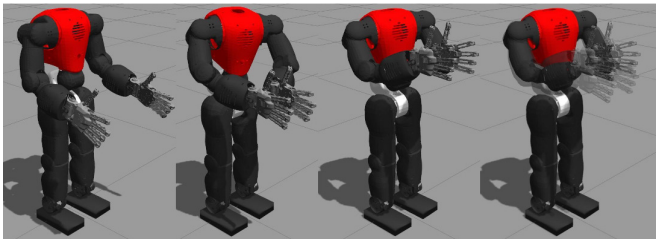

Fig. 2: Demonstrations of the bimanual shaking task in simulation. **First:** The robot is in a neutral starting pose. **Second:** Reaching for the bottle and grasping it. **Third:** Bringing the bottle close to the torso. **Fourth:** Shaking movement executed through rhythmic oscillations of both shoulder joints.

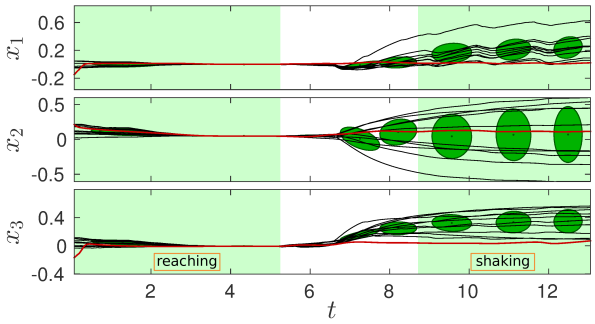

Fig. 3: Left end-effector position (in meters) with respect to the initial bottle coordinate system (prior to the grasp). Black lines represent demonstrations while the red line represents one reproduction. Ellipses depict the Gaussian components of the model (isocontour of one standard deviation). The shaded areas mark the duration of the reaching and shaking phases.

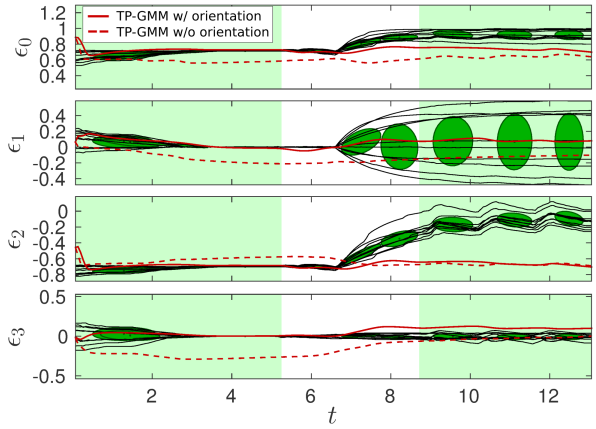

Fig. 4: Left end-effector orientation (as a unit quaternion) with respect to the initial bottle coordinate system (prior to the grasp). The solid red lines represent the reproduction using a TP-GMM that encodes both position and orientation. The dashed lines correspond to a reproduction with a TP-GMM that encodes only position.

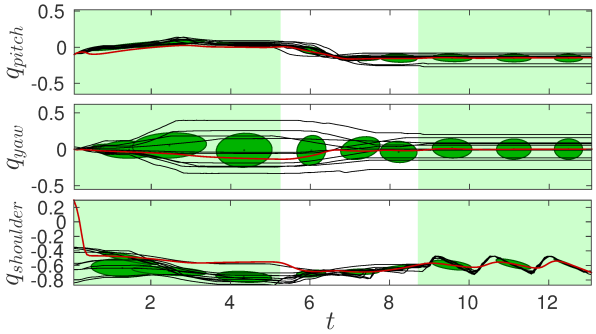

Fig. 5: Waist joints (pitch and yaw) and shoulder joint of the left arm (in radians). Notice the low variability of the shoulder joint during the shaking part (second shaded area) and how it is captured by the model.

The Jacobian matrix is given by [15], $\boldsymbol{J}_{\text{up}} = \begin{bmatrix} \boldsymbol{J}_{W|L} & \boldsymbol{J}_L & \boldsymbol{0} \\ \boldsymbol{J}_{W|R} & \boldsymbol{0} & \boldsymbol{J}_R \end{bmatrix}$, where $\boldsymbol{J}_{W|L}$, $\boldsymbol{J}_{W|R}$ denote the Jacobians that account for the effect of the waist joints on left and right end-effector velocities. $\boldsymbol{J}_L$ and $\boldsymbol{J}_R$ correspond to the Jacobians of the left and right end-effectors from the waist link. The minimum norm inverse kinematics solution is given in this case by $\hat{\dot{\boldsymbol{q}}} = \begin{bmatrix} \hat{\dot{\boldsymbol{q}}}_W^\mathsf{T} & \hat{\dot{\boldsymbol{q}}}_L^\mathsf{T} & \hat{\dot{\boldsymbol{q}}}_R^\mathsf{T} \end{bmatrix}^\mathsf{T} = \boldsymbol{J}_{\text{up}}^\dagger \begin{bmatrix} \dot{\boldsymbol{x}}_L^\mathsf{T} & \boldsymbol{\omega}_L^\mathsf{T} & \dot{\boldsymbol{x}}_R^\mathsf{T} & \boldsymbol{\omega}_R^\mathsf{T} \end{bmatrix}^\mathsf{T}$, where $\boldsymbol{J}_{\text{up}}^\dagger$ is the right pseudo-inverse of $\boldsymbol{J}_{\text{up}}$.

We collected 10 demonstrations of the skill, each of them with different initial bottle poses and a duration of approximately 13 seconds. In each demonstration, we recorded both end-effector poses with respect to the initial bottle frame and joint angles. Temporal alignment of the demonstrations was achieved using Dynamic Time Warping [16]. We used a TP-GMM with $K = 10$ components, chosen empirically. For this problem we considered $P = 2$ projection operators. The first operator is a concatenation of (6) and (15), for position and orientation constraints, parameterized with the initial pose of the bottle $\{\boldsymbol{p}_t^{(1)}, \boldsymbol{R}_t^{(1)}, \boldsymbol{\epsilon}_t^{(1)}\}$,

$$\boldsymbol{A}_t^{(1)} = \boldsymbol{J}_{\text{up}}^\dagger \begin{bmatrix} \boldsymbol{R}_t^{(1)} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \bar{\boldsymbol{H}}^*(\bar{\boldsymbol{\epsilon}}_{L,t-1}) \overset{+}{\bar{\boldsymbol{H}}}(\boldsymbol{\epsilon}_t^{(1)}) & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{R}_t^{(1)} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \bar{\boldsymbol{H}}^*(\bar{\boldsymbol{\epsilon}}_{R,t-1}) \overset{+}{\bar{\boldsymbol{H}}}(\boldsymbol{\epsilon}_t^{(1)}) \end{bmatrix}, \quad (16)$$

$$\boldsymbol{b}_t^{(1)} = \boldsymbol{J}_{\text{up}}^\dagger \begin{bmatrix} \boldsymbol{p}_t^{(1)} - \boldsymbol{x}_{L,t-1} \\ \boldsymbol{0} \\ \boldsymbol{p}_t^{(1)} - \boldsymbol{x}_{R,t-1} \\ \boldsymbol{0} \end{bmatrix} + \boldsymbol{q}_{t-1}. \quad (17)$$

The second projection operator is the canonical one, $\boldsymbol{A}_t^{(2)} = \boldsymbol{I}, \boldsymbol{b}_t^{(2)} = \boldsymbol{0}$. Subscripts L and R denote left and right end-effectors.

B. Results

We employed the learned TP-GMM to generate joint references $\hat{\boldsymbol{q}}_t$, for every time step of the reproduction, that were fed to a joint position controller, as per Algorithm 1. Figures 3-5 show the demonstration data over time† (black lines), in operational and configuration spaces, together with the Gaussian components obtained after EM (green ellipses). In addition we also plot the references generated by GMR (red lines), for a new position and orientation of the bottle. In Figures 3 and 4 we see that, during the reach and grasp movement, there is low variability in the demonstrations, both in position and orientation, when the end-effector is touching the bottle ($t \approx 4s$). This is successfully encoded by the model (narrow Gaussians showing low variance), as this aspect of the skill is important for a correct completion of the task. It follows that the synthesized movement (solid red line) closely matches the demonstrations in the regions of low variability. Note that, after the grasp ($t > 7s$), the variance increases as the end-effectors move away from the initial bottle pose to perform the shaking movement. In Figure 4, the dashed red line corresponds to a different reproduction of the task using a TP-GMM that does not encode orientation constraints. The resulting curves strongly differ

†Due to space limitations we only plot data corresponding to the left arm.

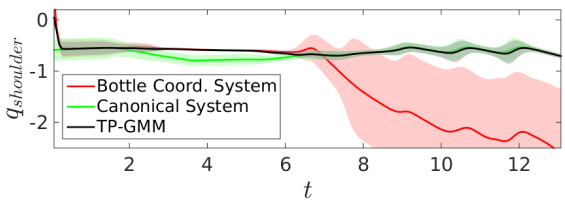

Fig. 6: Shoulder joint angle estimation (radians) from each space (red and green) and the resulting reference after TP-GMM (black). Each estimate has an associated mean (solid line) and variance (light color envelope), learned from demonstrations and synthesized during reproduction.

from the demonstrations, attesting that orientation encoding is indeed essential in this task (i.e. inadequate orientations result in unsuccessful grasps). Figure 5 shows that, from the beginning of the shaking phase ($t \approx 8s$), the shoulder joint (bottom graph) exhibits a consistent oscillatory pattern that is modeled by 3 Gaussians, which is adequately captured and synthesized by the model. This contrasts with the other joints, which do not influence the shaking. When using the novel task parameters, each candidate projection operator corresponds to a possible configuration space solution. The weight of each solution is estimated from the demonstrations, based on the variability in the data and the covariance matrices of the Gaussian components encoding it. Figure 6 shows that TP-GMM correctly extracted the most relevant solution according to the requirements of the task. Notice how, until $t \approx 5s$, TP-GMM (black line) matches the candidate solution given by the bottle coordinate system (red line). Since the variability encoded in this coordinate system is low compared to that of the configuration space (because reaching and grasping is done in operational space), the Gaussian product (2) favors this solution. This is achieved through the linear transformation properties of Gaussians, that allow for both centers and covariance matrices to be locally mapped from operational to configuration space using the proposed linear operators. Similarly, during the shaking phase, after $t \approx 8s$, the reference generated for the shoulder joint matches the solution obtained using the canonical projection operator. This is because the shaking movement results in a consistent oscillatory pattern of shoulder joints, observed during the demonstrations as seen in Figure 5. These results show that the proposed TP-GMM formulation is a viable solution for encoding relevant task features in both configuration and operational spaces, including orientation.

Finally, Figure 1 shows the two distinct phases of the movement during a reproduction in the real COMAN platform. In this experiment, we used a tray to carry a shaker towards COMAN, where an optical tracking system provided the initial shaker pose to the robot. In the top row of Figure 1, the robot takes into account the operational space constraints as it reaches for the bottle, while in the bottom row, the robot shakes the grasped bottle with rhythmic shoulder movements. In both parts of the movement, the operational and configuration space constraints are properly replicated.

V. CONCLUSIONS

We presented a framework for human-robot transfer of bimanual skills based on TP-GMM, which permits the consid-

eration of constraints in configuration and operational spaces, including orientation. The approach was validated in the COMAN robot which learned a bimanual task that required motion patterns in both spaces, for a proper execution.

In future work we will investigate equivalent formulations of orientation operators for torque controllers. Torque control is present in most modern robots and it has been shown in [17] that it allows for teaching force constraints to robots, in addition to kinematic ones such as those considered here. Finally, we plan to study how the proposed approach can be combined with Riemannian formulations [18] for learning end-effector poses. While in our work the spherical geometry of $\mathcal{S}^3$ is approximated by the K Gaussians in the TP-GMM, such formulations may allow for a more precise encoding of orientation constraints by accounting for the unit norm constraint.

APPENDIX: UNIT QUATERNIONS

A unit quaternion $\boldsymbol{\epsilon} \in \mathcal{S}^3$ is defined by $\boldsymbol{\epsilon} = [\epsilon_0 \ \epsilon_1 \ \epsilon_2 \ \epsilon_3]^\mathsf{T} = \begin{bmatrix} v & \boldsymbol{u}^\mathsf{T} \end{bmatrix}^\mathsf{T}$, where $v \in \mathbb{R}$ and $\boldsymbol{u} \in \mathbb{R}^3$, following the notation used by [14], are the real and vector parts of the quaternion. The conjugate of a unit quaternion is denoted by $\bar{\boldsymbol{\epsilon}} = \begin{bmatrix} u & -\boldsymbol{u}^\mathsf{T} \end{bmatrix}^\mathsf{T}$. As the name implies, unit quaternions have unitary norm, i.e., $v^2 + \boldsymbol{u}^\mathsf{T}\boldsymbol{u} = \epsilon_0^2 + \epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2 = 1$.

Composition of unit quaternions: Similarly to the product between rotation matrices, the quaternion product is non-commutative. It is defined by

$$\boldsymbol{\epsilon}_1 * \boldsymbol{\epsilon}_2 = \begin{bmatrix} v_1 v_2 - \boldsymbol{u}_1^\mathsf{T}\boldsymbol{u}_2 \\ v_1\boldsymbol{u}_2 + v_2\boldsymbol{u}_1 + \boldsymbol{u}_1 \times \boldsymbol{u}_2 \end{bmatrix}, \tag{18}$$

and it can be interpreted as a rotation operator: it rotates the frame whose orientation is described by $\boldsymbol{\epsilon}_2$ by the rotation defined by $\boldsymbol{\epsilon}_1$. Moreover, the quaternion product $\boldsymbol{\epsilon}_1 * \bar{\boldsymbol{\epsilon}}_2$ yields the quaternion that rotates $\boldsymbol{\epsilon}_2$ into $\boldsymbol{\epsilon}_1$.

Quaternion matrix: The product between two quaternions $\boldsymbol{\alpha} = [\alpha_0 \ \alpha_1 \ \alpha_2 \ \alpha_3]^\mathsf{T}$ and $\boldsymbol{\beta} = [\beta_0 \ \beta_1 \ \beta_2 \ \beta_3]^\mathsf{T}$ can also be written in matrix form by resorting to Hamilton operators (quaternion matrices):

$$\boldsymbol{\alpha} * \boldsymbol{\beta} = \overset{+}{\boldsymbol{H}}(\boldsymbol{\alpha})\boldsymbol{\beta} = \bar{\boldsymbol{H}}(\boldsymbol{\beta})\boldsymbol{\alpha}, \tag{19}$$

with Hamilton operators $\overset{+}{\boldsymbol{H}}$, $\bar{\boldsymbol{H}}$ defined by (see also [19])

$$\overset{+}{\boldsymbol{H}}(\boldsymbol{\alpha}) = \begin{bmatrix} \alpha_0 & -\alpha_1 & -\alpha_2 & -\alpha_3 \\ \alpha_1 & \alpha_0 & -\alpha_3 & \alpha_2 \\ \alpha_2 & \alpha_3 & \alpha_0 & -\alpha_1 \\ \alpha_3 & -\alpha_2 & \alpha_1 & \alpha_0 \end{bmatrix}, \bar{\boldsymbol{H}}(\boldsymbol{\beta}) = \begin{bmatrix} \beta_0 & -\beta_1 & -\beta_2 & -\beta_3 \\ \beta_1 & \beta_0 & \beta_3 & -\beta_2 \\ \beta_2 & -\beta_3 & \beta_0 & \beta_1 \\ \beta_3 & \beta_2 & -\beta_1 & \beta_0 \end{bmatrix}. \tag{20}$$

Notice the commutativity between $\overset{+}{\boldsymbol{H}}$ and $\bar{\boldsymbol{H}}$ in (19). Even though the quaternion product is non-commutative, Hamilton operators commute between them. This result is useful when we want to change the order of the quaternions being multiplied without affecting the resulting orientation.

REFERENCES

[1] C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D. V. Dimarogonas, and D. Kragic, "Dual arm manipulation – a survey," *Robotics and Autonomous Systems*, vol. 60, no. 10, pp. 1340 – 1353, 2012.

[2] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, May 2009.

[3] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, "Statistical dynamical systems for skills acquisition in humanoids," in *Proc. IEEE-RAS Intl Conf. on Humanoid Robots (Humanoids)*, Osaka, Japan, November-December 2012, pp. 323–329.

[4] J. Umlauft, D. Sieber, and S. Hirche, "Dynamic movement primitives for cooperative manipulation and synchronized motions," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Hong Kong, China, May-June 2014, pp. 766–771.

[5] A. L. P. Ureche and A. Billard, "Encoding bi-manual coordination patterns from human demonstrations," in *Proc. ACM/IEEE Intl Conf. on Human-Robot Interaction (HRI)*, Bielefeld, Germany, March 2014, pp. 264–265.

[6] R. Lioutikov, O. Kroemer, J. Peters, and G. Maeda, "Learning manipulation by sequencing motor primitives with a two-armed robot," in *Proc. Intl Conf. on Intelligent Autonomous Systems (IAS)*, ser. Advances in Intelligent Systems and Computing, vol. 302. Springer, 2014.

[7] N. Likar, B. Nemec, L. Zlajpah, S. Ando, and A. Ude, "Adaptation of bimanual assembly tasks using iterative learning framework," in *Proc. IEEE-RAS Intl Conf. on Humanoid Robots (Humanoids)*, Seoul, South Korea, November 2015, pp. 771–776.

[8] 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, September–October 2015, pp. 464–470.

[9] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[10] S. Calinon and A. G. Billard, "Statistical learning by imitation of competing constraints in joint space and task space," *Advanced Robotics*, vol. 23, no. 15, pp. 2059–2076, 2009.

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

[12] N. G. Tsagarakis, S. Morfey, G. A. Medrano-Cerda, Z. Li, and D. G. Caldwell, "COMpliant huMANoid COMAN: Optimal joint stiffness tuning for modal frequency control." in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Karlsruhe, Germany, May 2013, pp. 673–678.

[13] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer, 2009.

[14] 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.

[15] J. Lee, A. Ajoudani, E. M. Hoffman, A. Rocchi, A. Settimi, M. Ferrati, A. Bicchi, N. G. Tsagarakis, and D. G. Caldwell, "Upper-body impedance control with variable stiffness for a door opening task," in *Proc. IEEE-RAS Intl Conf. on Humanoid Robots (Humanoids)*, Madrid, Spain, November 2014, pp. 713–719.

[16] C. Y. Chiu, S. P. Chao, M. Y. Wu, and S. N. Yang, "Content-based retrieval for human motion data," *Visual Communication and Image Representation*, vol. 15, pp. 446–466, 2004.

[17] J. Silvério, Y. Huang, L. Rozo, S. Calinon, and D. G. Caldwell, "Probabilistic learning of torque controllers from kinematic and force constraints," in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Madrid, Spain, Oct. 2018.

[18] 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.

[19] B. Adorno, "Two-arm manipulation: From manipulators to enhanced human-robot collaboration," Ph.D. dissertation, Université Montpellier 2, 2011.