Safe Execution of Learned Orientation Skills with Conic Control Barrier Functions

Zheng Shen[1], Matteo Saveriano[2], Fares J. Abu-Dakka[3], and Sami Haddadin[1]

Abstract— In the field of Learning from Demonstration (LfD), Dynamical Systems (DSs) have gained significant attention due to their ability to generate real-time motions and reach predefined targets. However, the conventional convergence-centric behavior exhibited by DSs may fall short in safety-critical tasks, specifically, those requiring precise replication of demonstrated trajectories or strict adherence to constrained regions even in the presence of perturbations or human intervention. Moreover, existing DS research often assumes demonstrations solely in Euclidean space, overlooking the crucial aspect of orientation in various applications. To alleviate these shortcomings, we present an innovative approach geared toward ensuring the safe execution of learned orientation skills within constrained regions surrounding a reference trajectory. This involves learning a stable DS on $SO(3)$, extracting time-varying conic constraints from the variability observed in expert demonstrations, and bounding the evolution of the DS with Conic Control Barrier Function (CCBF) to fulfill the constraints. We validated our approach through extensive evaluation in simulation and showcased its effectiveness for a cutting skill in the context of assisted teleoperation.
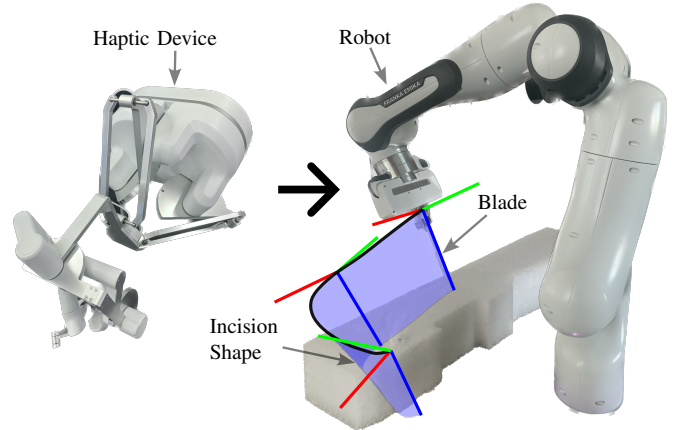
Fig. 1. Setup of the experiment for assisted teleoperation. A human operator uses a haptic device to adjust the cutting skill executed by a robot. The z-axis of the EE frame (blue segment) coincides with the blade. The shaded blue surface indicates the incision shape.

I. INTRODUCTION

LfD enables robots to learn novel skills by imitating human actions instead of coding them [1]. This involves automatic extraction task requirements from human demonstrations. Ideally, the acquired skills are generalizable, agnostic to specific robot platforms, and robust against perturbations. Among LfD approaches, DSs are attractive due to their capability to generate real-time motions and converge toward a predefined target [2]. During the execution phase, a robot's initial state is fed into the DS, enabling the robot to navigate and reach its intended goal despite environmental uncertainties or changes. Nevertheless, while this convergence-centric behavior proves effective in various applications, it may be inadequate or even counterproductive in safety-critical tasks that demand the robot to closely mimic the demonstrated trajectories or stay strictly within a constrained region [3].

The majority of works on DSs assume demonstrations in Euclidean space and primarily consider task constraints like obstacle avoidance concerning safety. However, it is clear that orientation, residing in non-Euclidean space, also plays a significant role in various applications. For instance, consider the task of transporting and pouring a cup of water. Here, it becomes crucial to constrain the orientation of the cup to prevent unintended spillage, a requirement that goes beyond the conventional scope of Euclidean task constraints. Similarly, for tissue resection in the domain of robotic surgery, precise orientation of surgical instruments is vital for performing delicate procedures accurately and safely. Indeed, as shown in Fig. 1, the orientation trajectory directly influences the shape of the incision.

To address these challenges, we introduce a novel approach that ensures the safe execution of acquired orientation skills when subjected to perturbation or human intervention. This approach guarantees that the execution stays close to a reference trajectory within the region defined by the constraints derived from the variability of demonstrations. The primary contributions of this work are: (*i*) An extension of Physically-Consistent Gaussian Mixture Model (PC-GMM) [4] for learning stable DSs on $SO(3)$. Notably, we provide a formal proof of its asymptotic stability. (*ii*) A novel approach that extracts orientation constraints based on the variability of demonstrations and exploits Conic Control Barrier Function (CCBF) [5] to ensure safe execution. (*iii*) Validation of our method in simulation and in an assisted teleoperation experiment focused on robotic cutting tasks.

II. RELATED WORK

A significant body of LfD literature focuses on stable motion execution. One of the most prevalent methods, the

[1]Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich, Germany.

[2]Department of Industrial Engineering (DII), University of Trento, Trento, Italy.

[3]Electronic and Informatics Department, Faculty of Engineering, Mondragon Unibertsitatea, Bilbao, Spain.

Corresponding author: zheng.shen@tum.de

Dynamic Motion Primitive (DMP) [6], [7], employs a phase variable to cancel the nonlinear forcing term superimposed on a stable linear DS to ensure convergence. DMP, being a deterministic method, is limited to learning from a single demonstration. The Stable Estimator of Dynamical Systems (SEDS) [2] approximates a nonlinear DS with a combination of linear ones but faces accuracy and stability dilemmas. PC-GMM [4] alleviates this limitation by decoupling the parameters of Gaussian Mixture Model (GMM) from those of the linear DSs. The DS formulation in [3] provides a behavior akin to "trajectory-tracking" for specific reference trajectories and retains global convergence. Although the aforementioned approaches are effective in learning position trajectories, they neglect orientation. In [8], Riemannian metrics are used to learn orientation trajectories via task-parameterized GMM, without considering stability. Abu-Dakka et al. [9] extended DMP to Riemannian manifolds like quaternions, while Saveriano et al. [10] learnd a diffeo-morphism to map simple manifold trajectories into complex ones. In our current work, we draw inspiration from the supe-rior performance of PC-GMM [4] and extend the framework to orientation and establish its asymptotic stability on $SO(3)$.

Learning constraints from demonstrations have been ex-plored in the literature. In [11], position and force constraints are extracted from demonstration variance. Menner et al. [12] learned cost functions and constraints simultaneously in an inverse optimal control framework, but the constraint is limited to a convex hull in Euclidean space. In [13], linear subspaces are learned as constraints incrementally and used Control Barrier Functions (CBFs) to generate constrained motions. CBFs were also used in [14] to maintain a joint space trajectory within a learned covariance bound. A neural network [15] and kernelized principle component analy-sis [16] are used to learn equality constraints as manifolds, whereas the learned constraints are on feature space and lack of explainability. Perez-D'Arpino et al. [17] utilized task space region, a volume in $SE(3)$ around the keyframes, for constrained motion planning. Chou et al. [18] proposed a method to learn grid and parametric constraints from demon-strations and uses Euler angles to represent the orientation. Like most works, we exploit the variability in demonstrations and extract orientation constraints based on conic constraints.

CBFs render a set forward invariant and are thus widely used in the control of constrained systems [19]. Time-Varying Control Barrier Function (TV-CBF) [20] can be used to enforce time-varying constraints. Wu and Sreenath [21] extended CBFs from Euclidean space to manifolds with applications to geometrical control of mechanical systems under time-varying constraints. In [5], CCBFs are used to solve a distributed collision avoidance problem for a group of agents on a sphere. Tan et al. [22] proposed a union of hyperspherical constraints as CBF on $SO(3)$ and applied it to safety-critical control along a $\mathcal{C}^2$ reference trajectory. In comparison, a conic constraint is preferred in this work as it provides more flexibility when constructing constraints.

In assisted teleoperation, robots aid in task completion and simplify teleoperation for the human operator. The most prevalent method is virtual fixture [23], which assists the operator by guiding or constraining either the user's input or the robot's motion. Ewerton et al. [24] provided haptic cues for goal-reaching tasks by constructing a potential field based on learned GMM over demonstrations. Dragan and Srinivasa [25] formalized assisted teleoperation as a policy blending problem, arbitrating human input and robot action based on the confidence of prediction of human intention. The prediction of intended goals was further improved by solving a partially observable Markov decision process with hindsight optimization [26]. Task-Parameterized Gaussian Mixture Model (TP-GMM) encodes demonstrated trajectories and blends the learned model and user inputs with a linear quadratic regulator [27]. In [28], the remote robot autonomously executed a task learned with a task-parameterized hidden GMM, with the user only providing high-level task goals. Mower et al. [29] proposed a receding horizon shared control method, which adapts future trajec-tories based on the estimation of the operator's intended skill. We show the applicability of our method to assisted teleoperation while providing formal safety guarantees.

III. PRELIMINARIES

A. Special Orthogonal Group $SO(3)$

Traditionally, orientations in 3D-space are represented as rotation matrices that inhabit the Special Orthogonal Group denoted as $SO(3)$ [30], where

$$SO(3) := \left\{ \mathbf{R} \in \mathbb{R}^{3\times3} \mid \mathbf{R}^\top \mathbf{R} = \mathbf{R}\mathbf{R}^\top = \mathbf{I}, \det(\mathbf{R}) = 1 \right\}.$$

The group equipped with matrix multiplication constitutes a Lie Group. The associated Lie algebra comprises all 3×3 skew-symmetric matrices, i.e., $\mathfrak{so}(3) := \left\{ \mathbf{\Omega} \in \mathbb{R}^{3\times3} \mid \mathbf{\Omega}^\top = -\mathbf{\Omega} \right\}$. The mapping $(\cdot)^\wedge$, also denoted as $[(\cdot)]_\times : \mathbb{R}^3 \to \mathfrak{so}(3)$, along with its inverse mapping $(\cdot)^\vee : \mathfrak{so}(3) \to \mathbb{R}^3$ are defined as

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \underset{(\cdot)^\vee}{\overset{[(\cdot)]_\times}{\rightleftharpoons}} [\mathbf{x}]_\times = \begin{pmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{pmatrix}.$$

Throughout the paper, we refer to $\mathbf{x}$ such that $[\mathbf{x}]_\times \in \mathfrak{so}(3)$ as a tangent vector.

For a given $[\mathbf{x}]_\times \in \mathfrak{so}(3)$, the exponential map function $\exp(\cdot) : \mathfrak{so}(3) \to SO(3)$ allows the representation of $[\mathbf{x}]_\times$ as rotation matrices

$$\exp([\mathbf{x}]_\times) = \begin{cases} \mathbf{I} + \frac{\sin(\|\mathbf{x}\|)}{\|\mathbf{x}\|}[\mathbf{x}]_\times + \frac{1-\cos(\|\mathbf{x}\|)}{\|\mathbf{x}\|^2}[\mathbf{x}]_\times^2, & \mathbf{x} \neq 0 \\ \mathbf{I}, & \text{otherwise.} \end{cases} \tag{1}$$

Its inverse is the logarithmic map $\log(\cdot) : SO(3) \to \mathfrak{so}(3)$

$$\log(\mathbf{R}) = \begin{cases} \frac{\theta(\mathbf{R})}{2\sin(\theta(\mathbf{R}))} \left(\mathbf{R} - \mathbf{R}^\top \right), & \mathbf{R} \neq \mathbf{I} \\ \mathbf{0}, & \text{otherwise,} \end{cases} \tag{2}$$

where $\theta(\mathbf{R}) := \arccos((\text{tr}(\mathbf{R}) - 1)/2)$. To simplify the notation, we introduce the uppercase exponential map as $\text{Exp}(\mathbf{x}) := \exp([\mathbf{x}]_\times)$ and the uppercase logarithmic map as $\text{Log}(\mathbf{R}) := (\log(\mathbf{R}))^\vee$. Additionally, we define $\text{Log}_{\mathbf{R}_b}(\mathbf{R}_a) := \text{Log}(\mathbf{R}_a^\top \mathbf{R}_b)$.

B. Time-Varying Control Barrier Function

Consider the control-affine system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \tag{3}$$

where $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n$ is the state, $\mathbf{u} \in \mathcal{U} \subset \mathbb{R}^m$ is the control input, and $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ and $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^{n \times m}$ are locally Lipschitz continuous functions.

Definition 1 (Extended class $\mathcal{K}_\infty$ function): A continuous function $\alpha : (-b, a) \to \mathbb{R}$ belongs to the extended class $\mathcal{K}_\infty$ for some $a, b > 0$, if $\alpha(0) = 0$ and α increases strictly monotonically.

Definition 2 (forward invariance of time-varying set) A time-varying set $\mathcal{C}(t) \subset \mathbb{R}^n$ is forward invariant to (3) for a given control law $\mathbf{u}$, if for any $\mathbf{x}_0 \in \mathcal{C}(t_0)$, there exists a unique solution $\boldsymbol{\phi} : [t_0, t_1] \to \mathbb{R}^n$ with $\boldsymbol{\phi}(t_0) = \mathbf{x}_0$ and $\frac{d}{dt}\boldsymbol{\phi}(t) = \mathbf{f}(\boldsymbol{\phi}(t)) + \mathbf{g}(\boldsymbol{\phi}(t))\mathbf{u}$ such that $\boldsymbol{\phi}(t) \in \mathcal{C}(t)$ for all $t \in [t_0, t_1]$.

Definition 3 (time-varying control barrier function): Let $\mathcal{C}(t) := \{\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n \mid h(\mathbf{x}, t) \geq 0\}$ be the 0-superlevel set of a smooth function $h(\mathbf{x}, t) : \mathbb{R}^n \times [t_0, t_1] \to \mathbb{R}$, then h is a TV-CBF if there exist an extended class $\mathcal{K}_\infty$ function α such that for all $(\mathbf{x}, t) \in \mathcal{X} \times [t_0, t_1]$, the control system in (3) satisfies

$$\sup_{\mathbf{u} \in \mathcal{U}} \left[L_\mathbf{f} h(\mathbf{x}, t) + L_\mathbf{g} h(\mathbf{x}, t)\mathbf{u} \right] + \frac{\partial h(\mathbf{x}, t)}{\partial t} \geq -\alpha(h(\mathbf{x}, t)). \tag{4}$$

Applying any Lipschitz continuous controller $u(\mathbf{x}, t) \in \mathcal{U}(\mathbf{x}, t)$ to (3), $\mathcal{U}(\mathbf{x}, t) = \{\mathbf{u} \in \mathcal{U} \mid L_\mathbf{f} h(\mathbf{x}, t) + L_\mathbf{g} h(\mathbf{x}, t)\mathbf{u} + \frac{\partial h(\mathbf{x}, t)}{\partial t} + \alpha(h(\mathbf{x}, t)) \geq 0\}$, the forward invariance of $\mathcal{C}(t)$ is guaranteed [31, Theorem 1], which enables us to use the following Quadratic Programming (QP) that minimally modifies the reference controller $\mathbf{u}_0(\mathbf{x}, t)$:

$$\mathbf{u}^*(\mathbf{x}, t) = \operatorname*{argmin}_{\mathbf{u} \in \mathcal{U}} \frac{1}{2}\|\mathbf{u} - \mathbf{u}_0(\mathbf{x}, t)\|_2^2 \text{ subject to Eq. (4).} \tag{5}$$

C. Conic Control Barrier Function

Similarly to the case of Euclidean space, we can define a CBFs on $SO(3)$ as follows:

Definition 4 (CBF on $SO(3)$): Consider the 0-superlevel set $\mathcal{C} = \{\mathbf{R} \in \mathcal{S} \subset SO(3) \mid h(\mathbf{R}) \geq 0\}$ of a smooth function $h(\mathbf{R}) : SO(3) \to \mathbb{R}$, then h is a CBF on $SO(3)$ if there exists an extended class $\mathcal{K}_\infty$ function α such that for all $\mathbf{R} \in \mathcal{S}$ the attitude dynamics $\dot{\mathbf{R}} = \mathbf{R}[\boldsymbol{\omega}]_\times$ satisfies

$$\sup_{\boldsymbol{\omega} \in \boldsymbol{\Omega}} \left(\dot{h}(\mathbf{R}) \right) \geq -\alpha \left(h(\mathbf{R}) \right) \tag{6}$$

One option for $h(\mathbf{R})$ is conic attitude constraints. Consider the inequality constraint as follows:

$$h_i(\mathbf{R}) := \mathbf{e}_i^\top \mathbf{R} \mathbf{e}_i - \cos(\theta_i) \geq 0, \tag{7}$$

where $\theta_i \in (0, \pi/2)$ determines the size of the cone and $\mathbf{e}_i = [\delta_{i1}\ \delta_{i2}\ \delta_{i3}]^\top$, with δ_{ij} being the Kronecker delta and $i, j \in \{1, 2, 3\}$. As illustrated in Fig.2, the transformed axis $\mathbf{R}\mathbf{e}_i$ is constrained in the conic region determined by $\mathbf{e}_i$ and θ_i. Ibuki et al. [5] proves that for the attitude dynamics, any Lipschitz continuous control law $\boldsymbol{\omega}(\mathbf{R}) \in \boldsymbol{\Omega}(\mathbf{R})$, $\boldsymbol{\Omega}(\mathbf{R}) =$
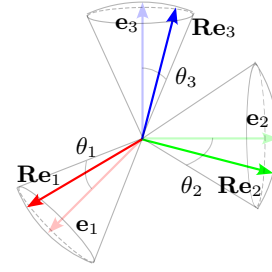

Fig. 2. An illustration of conic constraints.

$\{\boldsymbol{\omega} \in \boldsymbol{\Omega} \mid -\mathbf{e}_i^\top \mathbf{R}[\mathbf{e}_i]_\times \boldsymbol{\omega} + \alpha(\mathbf{e}_i^\top \mathbf{R}\mathbf{e}_i - \cos(\theta_i)) \geq 0\}$ will render the set $\mathcal{C}$ forward invariant[1].

IV. METHODOLOGY

Our method comprises two phases: an offline phase dedicated to skill and constraint acquisition from demonstrations, and an online phase focused on the safe execution of these skills within the regions defined by the constraints. Fig. 3 provides an overview of the method.

During the learning phase, PC-GMM encodes translational and rotational motions as two stable DSs, extracting time-varying conic constraints for orientation based on observed demonstration variability. Specifically, the cone axes align with the learned DS, with cone angles $\boldsymbol{\theta}$ representing orientation variances encoded in a Locally Weighted Regression (LWR) model. In the execution phase, each iteration adjusts the command $\boldsymbol{\omega}_{\text{exc}}$ from the learned DS with human input $\mathbf{u}_h$ or perturbed by noise $\mathbf{u}_p$. The summation $\mathbf{u}_0$ serves as the reference control input for the Conic Control Barrier Function Quadratic Programming (CCBF-QP), which acts as a filter that minimally modifies the given input while guaranteeing forward invariance of the safety set defined by the time-varying conic constraints. Finally, we obtain the trajectory to be executed $\mathbf{R}_{\text{exc}}(t)$ by discretely integrating the solution $\mathbf{u}^*$ to the QP.
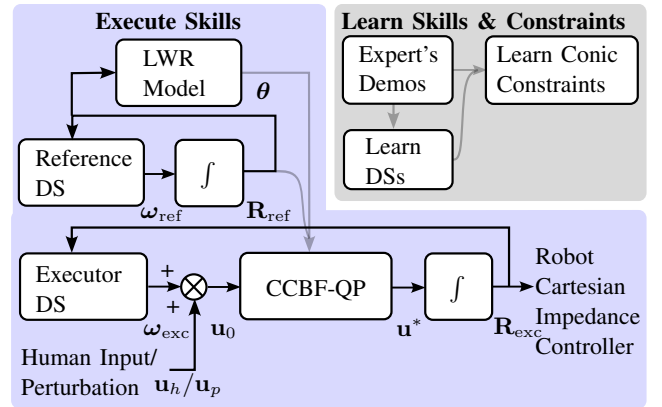

Fig. 3. Overview of the method. Dark-colored lines indicate values. Light-colored lines indicate data or functions.

---

[1]The property $[\mathbf{a}]_\times \mathbf{b} = -[\mathbf{b}]_\times \mathbf{a}$ is used to obtain the CCBF condition.

A. Learning from Demonstration

We formulate robot motions $(\mathbf{p}, \mathbf{R}) \in \mathbb{R}^3 \times SO(3)$ as a control law driven by two separate DSs that encode a specific behavior. From a machine learning perspective, estimating the DSs from a set of N reference trajectories $\mathcal{D} = \left\{ \left(\mathbf{p}^{t,n}, \mathbf{R}^{t,n} \right), \left(\dot{\mathbf{p}}^{t,n}, \boldsymbol{\omega}^{t,n} \right) \right\}_{t=0,n=1}^{T^n,N}$ can be framed as a regression problem. In this work, we use PC-GMM to learn the DSs [4]. For translation, we use a first-order DS

$$\dot{\mathbf{p}} = \mathbf{f_p}(\mathbf{p}), \ \mathbf{f_p} : \mathbb{R}^3 \to \mathbb{R}^3. \tag{8}$$

For $\mathbf{R} \in SO(3)$, given the aim of modeling robots' motion, we assume they follow the rigid body motion [32]

$$\dot{\mathbf{R}} = \mathbf{R}[\boldsymbol{\omega}]_\times. \tag{9}$$

The capitalized logarithmic map allows the regression of rotation trajectories to stay in $\mathbb{R}^3$, with tangent vector as inputs and angular velocity $\boldsymbol{\omega}$ as outputs

$$\boldsymbol{\omega} = \mathbf{f_R}\left(\mathrm{Log}_{\mathbf{R}_g}(\mathbf{R}) \right), \ \mathbf{f_R} : \mathbb{R}^3 \to \mathbb{R}^3, \tag{10}$$

where $\mathbf{R}_g$ is the rotational goal.

Inspired by PC-GMM [4], we define the regression function in (10) as

$$\boldsymbol{\omega} = \sum_{k=1}^{K} \mathbf{A}_k \, \mathrm{Log}_{\mathbf{R}_g}(\mathbf{R}), \tag{11}$$

where $\mathbf{A}_k$ are positive definite matrices learned from demonstrations. In order to prove the stability of (11), we consider the trace Lyapunov candidate $V(\mathbf{R}) = \mathrm{tr}(\mathbf{I} - \mathbf{R}^\top \mathbf{R}_g)$, that is widely used in attitude control on $SO(3)$ [33]. The time derivative of $V(\mathbf{R})$ writes as:

$$\dot{V}(\mathbf{R}) = -\mathrm{tr}(\dot{\mathbf{R}}^\top \mathbf{R}_g) = -\mathrm{tr}([\boldsymbol{\omega}]_\times^\top \mathbf{R}^\top \mathbf{R}_g), \tag{12}$$

where we used the definition of $\dot{\mathbf{R}}$ in (9). Defining $\mathbf{E} = \mathbf{R}^\top \mathbf{R}_g$ and recalling the identity $\mathrm{tr}([\boldsymbol{\omega}]_\times^\top \mathbf{E}) = \boldsymbol{\omega}^\top \left[(\mathbf{E} - \mathbf{E}^\top) \right]^\vee$ [33], the derivative of the Lyapunov function in (12) becomes

$$
\begin{aligned}
\dot{V}(\mathbf{R}) &= -\boldsymbol{\omega}^\top \left[(\mathbf{E} - \mathbf{E}^\top) \right]^\vee \\
&= -\sum_{k=1}^{K} \left(\mathrm{Log}_{\mathbf{R}_g}(\mathbf{R}) \right)^\top \mathbf{A}_k \left[(\mathbf{E} - \mathbf{E}^\top) \right]^\vee,
\end{aligned} \tag{13}
$$

where we used the definition of $\boldsymbol{\omega}$ in (11) and the property $\mathbf{A}_k^\top = \mathbf{A}_k$. From the definition of $\mathrm{Log}_{\mathbf{R}_g}(\mathbf{R})$ in Sec. III-A, it is easy to verify that $\dot{V}(\mathbf{R})$ vanishes at the equilibrium, i.e., $\dot{V}(\mathbf{R}) = 0$ for $\mathbf{R} = \mathbf{R}_g$. For $\mathbf{R} \neq \mathbf{R}_g$, i.e., for $\mathbf{E} \neq \mathbf{I}$, it holds that

$$\mathrm{Log}_{\mathbf{R}_g}(\mathbf{R}) = \frac{\theta(\mathbf{E})}{2\sin(\theta(\mathbf{E}))} \left[(\mathbf{E} - \mathbf{E}^\top) \right]^\vee. \tag{14}$$

By substituting (14) into (13), we obtain that

$$\dot{V}(\mathbf{R}) = -\sum_{k=1}^{K} \frac{\theta(\mathbf{E})}{2\sin(\theta(\mathbf{E}))} \left(\left[(\mathbf{E} - \mathbf{E}^\top) \right]^\vee \right)^\top \mathbf{A}_k \left[(\mathbf{E} - \mathbf{E}^\top) \right]^\vee. \tag{15}$$

Recalling that $\mathbf{A}_k$ are positive definite matrices and that $\frac{\theta(\mathbf{E})}{\sin(\theta(\mathbf{E}))} \geq 0$ for $-\pi < \theta(\mathbf{E}) < \pi$ and it vanishes only at $\theta(\mathbf{E}) = 0$ (corresponding to $\mathbf{E} = \mathbf{I}$), we conclude the asymptotic stability of (11).

B. Learn time-varying conic constraints

We derive time-varying conic constraints, denoted as $h(\mathbf{R}, t)$, by independently learning cone axes and cone angles. We first outline the definition of the conic axes. By integrating Eq. (9) and Eq. (10) with a given initial point and goal, we reproduce the reference rotational trajectory $\mathbf{R}_{\mathrm{ref}}(t)$. Referring to the axes of the world frame as $\mathbf{e}_{1:3}^{\mathrm{w}}$, the cone axes are straightforwardly established as the axes of the transformed world frame, denoted as $\mathbf{e}_{1:3}^{\mathrm{ref}}(t) := \mathbf{R}_{\mathrm{ref}}(t)\mathbf{e}_{1:3}^{\mathrm{w}}$.

Algorithm 1 outlines the procedure to encode the variance of orientations as cone angles along demonstrated trajectories by learning the LWR model $[\theta_1, \theta_2, \theta_3]^\top = \mathbf{f}_\theta(\mathrm{Log}_{\mathbf{R}_g}(\mathbf{R}))$. It begins by computing the minimum distance D_{min} between the initial and goal rotations across all demonstrations. This distance is then uniformly partitioned to produce the dataset $\{d_i\}_{i=0}^{M}$. Subsequently, we resample the time-dependent demonstrations $\left\{ \mathbf{R}^{t,n} \right\}_{t=0,n=1}^{T^n,N}$ w.r.t. $\{d_i\}_{i=0}^{M}$. For each set of rotational matrices at the same distance, we calculate the mean and covariance of their tangent vectors. Projecting the sum of the mean and the covariance eigenvectors back onto $SO(3)$ yields the mean rotational matrix $\mathbf{R}_m$ and three additional rotational matrices $\mathbf{R}_{1:3}$. The cone angle for each axis $\mathbf{e}_i$ of a reference frame is defined as the maximum cosine angle between the transformed axes $\mathbf{R}_m\mathbf{e}_i$ and $\mathbf{R}_j\mathbf{e}_i$. Finally, we apply LWR to the dataset $\{d^i, \theta_{1:3}^i\}_{i=0}^{M}$ and obtain the regression model $\mathbf{f}(\boldsymbol{\theta})$. In conclusion, we establish the time-varying conic constraints as:

$$h_i(\mathbf{R}, t) := \mathbf{e}_i^{\mathrm{w}\top} \mathbf{R}^\top \mathbf{e}_i^{\mathrm{ref}}(t) - \cos(\theta_i(t)) \geq 0, i \in \{1,2,3\}, \tag{16}$$

with $\theta_i(t) = \mathbf{f}_{\theta i}(\| \mathrm{Log}_{\mathbf{R}_g}(\mathbf{R}_{\mathrm{ref}}(t)) \|_2)$.

Algorithm 1 LEARN CONE ANGLE

Require: $\left\{ \mathbf{R}^{t,n} \right\}_{t=0,n=1}^{T^n,N}$
1: $D_{min} = \min_{n=1}^{N} \| \mathrm{Log}_{\mathbf{R}_g^n}(\mathbf{R}^{0,n}) \|_2$
2: $d^i = (D_{\min}/M) \cdot i$ **for** $i = 1 : M$
3: $\left\{ \mathbf{R}^{i,n} \right\}_{i=0,n=1}^{M,N} \leftarrow \texttt{Resample}(\texttt{Slerp}(\{ \mathbf{R}^{t,n} \}_{t=0,n=1}^{T^n,N}))$
4: **for** $i \leq M$ **do**
5: $\quad \mathbf{m} \leftarrow \texttt{mean}(\left\{ \mathrm{Log}_{\mathbf{R}_g}(\mathbf{R}^{i,n}) \right\}_{n=1}^{N})$
6: $\quad \mathbf{V} \leftarrow \texttt{covariance}(\left\{ \mathrm{Log}_{\mathbf{R}_g}(\mathbf{R}^{i,n}) \right\}_{n=1}^{N})$
7: $\quad \mathbf{v}_{1:3} \leftarrow \texttt{Eigen\_Decomposition}(\mathbf{V})$
8: $\quad \mathbf{p}_{1:3} \leftarrow \mathbf{v}_{1:3} + \mathbf{m}$
9: $\quad \mathbf{R}_{1:3}, \mathbf{R}_m \leftarrow \mathrm{Exp}(\mathbf{p}_{1:3}), \mathrm{Exp}(\mathbf{m})$
10: $\quad \theta_{1:3}^i \leftarrow \max_{j \in \{1,2,3\}} \arccos(\mathbf{e}_{1:3}^{\mathrm{w}\top} \mathbf{R}_j^\top \mathbf{R}_m \mathbf{e}_{1:3}^{\mathrm{w}})$
11: **end for**
12: $\mathbf{f}_\theta \leftarrow \texttt{LWR}(\{d^i, \theta_{1:3}^i\}_{i=0}^{M})$
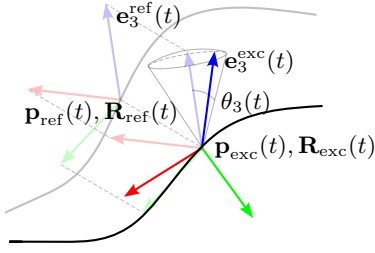13: **return** $\mathbf{f}_\theta$

Fig. 4. An illustration of time-varying conic constraints. The axes of a rotational matrix $\mathbf{R}_{\mathrm{exc}}$ always stay within the cones defined by $\mathbf{R}_{\mathrm{ref}}(t)$, which itself evolves according to the reference DS.

C. Constrained execution via CCBF-QP

The execution of skills is guaranteed to satisfy the time-varying conic constraints in Fig. 4 by solving (17). In Fig. 3, two distinct DSs run simultaneously: (*i*) the executor DS, which generates control input for the actively executed rotational trajectory $\mathbf{R}_{\mathrm{exc}}(t)$, and (*ii*) the reference DS, which provides the reference rotational trajectory $\mathbf{R}_{\mathrm{ref}}(t)$ for the time-varying conic constraints. Their respective axes are denoted as $\mathbf{e}_i^{\mathrm{exc}} = \mathbf{R}_{\mathrm{exc}}\mathbf{e}_i^{\mathrm{w}}$ and $\mathbf{e}_i^{\mathrm{ref}} = \mathbf{R}_{\mathrm{ref}}\mathbf{e}_i^{\mathrm{w}}$. At each time step, we compute angular velocities $\boldsymbol{\omega}_{\mathrm{ref}}$ and $\boldsymbol{\omega}_{\mathrm{exc}}$ using (10) and determine the cone angle $\boldsymbol{\theta}$ with the learned LWR model $\mathbf{f}_\theta$. The summation $\mathbf{u}_0 = \boldsymbol{\omega}_{\mathrm{exc}} + \mathbf{u}_h + \mathbf{u}_p$ serves as the reference control input to the QP, where $\mathbf{u}_h$ and $\mathbf{u}_p$ denote human inputs and perturbations. The reference trajectory follows exactly the learned DS $\dot{\mathbf{R}}_{\mathrm{ref}} = \mathbf{R}_{\mathrm{ref}}[\boldsymbol{\omega}_{\mathrm{ref}}]_\times$, while the execution follows $\dot{\mathbf{R}}_{\mathrm{exc}} = \mathbf{R}_{\mathrm{exc}}[\mathbf{u}^*]_\times$, where $\mathbf{u}^*$ is the solution to the following QP:

$$\underset{\mathbf{u}\in\mathcal{U}}{\operatorname{argmin}} \ \frac{1}{2}\|\mathbf{u} - \mathbf{u}_0(\mathbf{R}_{\mathrm{exc}}, t)\|_2^2 \qquad \text{(CCBF-QP)}$$
$$\text{s.t. } \dot{h}_i(\mathbf{R}_{\mathrm{exc}}, t) \geq -\alpha\left(h_i(\mathbf{R}_{\mathrm{exc}}, t)\right), i \in \{1, 2, 3\}. \tag{17}$$

The CCBF condition explicitly writes as

$$-\mathbf{e}_i^{\mathrm{ref}\top}\mathbf{R}_{\mathrm{exc}}\left[\mathbf{e}_i^{\mathrm{w}}\right]_\times \mathbf{u} - \mathbf{e}_i^{\mathrm{exc}\top}\mathbf{R}_{\mathrm{ref}}\left[\mathbf{e}_i^{\mathrm{w}}\right]_\times \boldsymbol{\omega}_{\mathrm{ref}}$$
$$+ \frac{d}{dt}\cos(\theta_i) + \alpha\left(\mathbf{e}_i^{\mathrm{ref}\top}\mathbf{e}_i^{\mathrm{exc}} - \cos(\theta_i)\right) \geq 0.$$

Notably, the executor DS can be arbitrarily selected since it will remain confined within the time-varying cones. For simplicity and more accurate reference trajectory tracking, we opt for the same executor DS as the reference DS. However, the reference DS evolving independently of human input could potentially pose challenges in assisted teleoperation, as the human operator may lose control authority over the execution. To address this, a human operator can modulate the actual velocity using an additional continuous input device, such as a foot pedal.

V. EXPERIMENTAL RESULTS

A. Simulation

In simulation, we used orientation trajectories from Riemannian LASA dataset (R-LASA) [10], an extension to LASA handwriting dataset [2]. We empirically set 5 Gaussian components for learning the translational and 8 for the rotational DSs. For LWR of cone angles, we opted for a
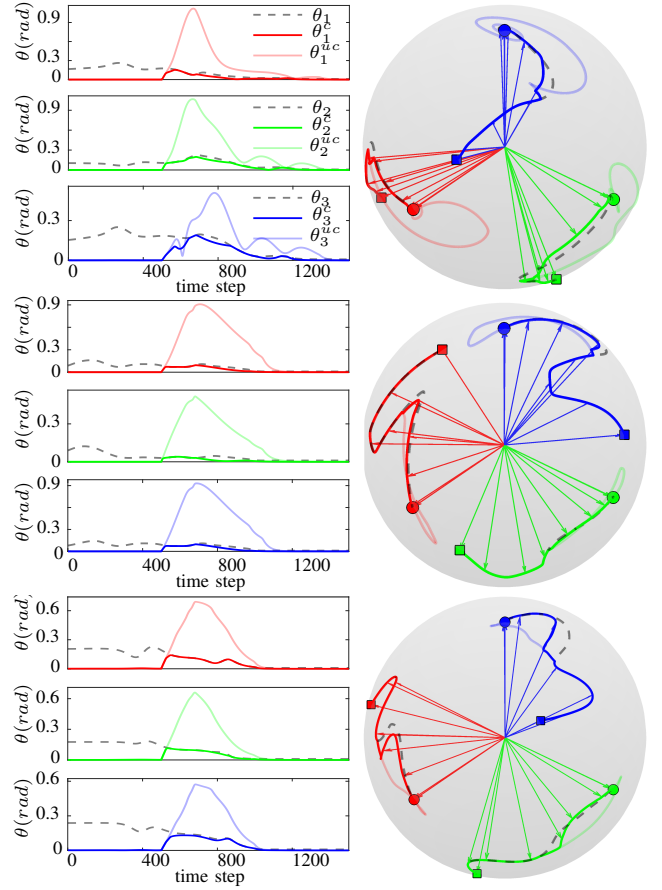

Fig. 5. Temporal evolution of cone angles and orientation trajectories for the L/N/W shapes (from top to bottom). Orientation is depicted in transformed frames, with square and circle markers denoting initial orientations and goals. On the left, deep/light solid lines represent the angles between constrained/unconstrained axes and reference axes, denoted as θ_i^c and θ_i^{uc} respectively. Red, green, and blue correspond to the x-axis, y-axis, and z-axis. Dashed black lines indicate the learned cone angles. On the right, the trajectories in solid deep/light blue correspond to constrained/unconstrained executions, and the dashed black trajectories represent the reference trajectories.

polynomial degree of 5 and employed 10 components, which appeared to offer well-balanced fitting accuracy. To avoid an empty admissible set, we artificially set the minimum value of the cone angle to $0.01\,\mathrm{rad}$. During execution, we employed a time step of $0.003\,\mathrm{s}$, starting at the average of the demonstrations and with the goal set as the identity matrix. Approximately $1.5\,\mathrm{s}$ after the start, we introduced a smooth step function lasting $0.5\,\mathrm{s}$ as the disturbance.

Fig. 5 illustrates the disturbed execution of the L/N/W shapes from the R-LASA. Firstly, we noticed that both trajectories with and without the CCBF-QP in all cases converged to the identity matrix, which verified the asymptotic stability of the DS learned with PC-GMM. However, all the unconstrained orientation trajectories deviated significantly from the reference trajectories after being disturbed, partly due to the inherent convergence behavior. In contrast, the constrained trajectories remained within the region defined by the time-varying cones, closely tracking the reference trajectories. To more clearly showcase the satisfaction of
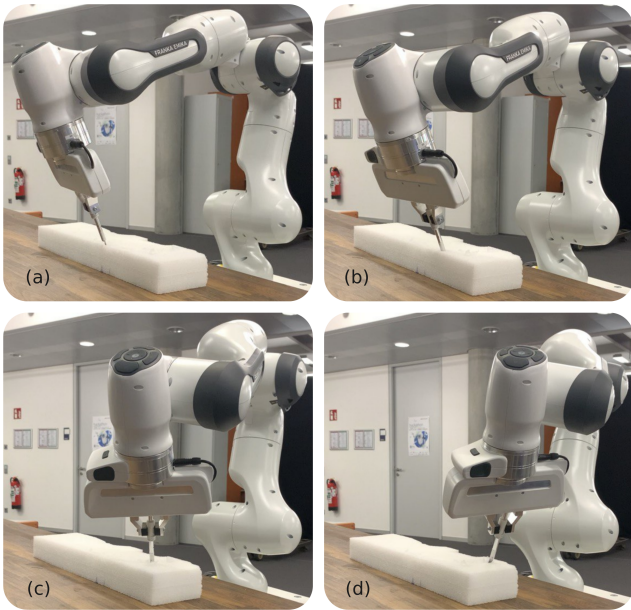
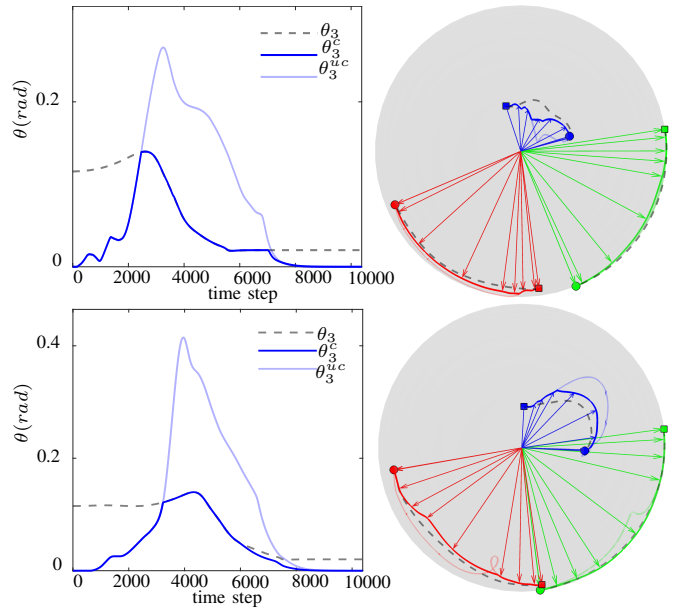Fig. 6. Execution of the cutting skill in assisted teleoperation.

Fig. 7. Cone angles over time and rotational trajectories of the EE frame of the robot. On the left, solid lines in deep/light blue represent the angles between constrained/unconstrained z-axes and reference z-axes, denoted as θ_3^c and θ_3^{uc} respectively, under human intervention. The black dashed lines indicate the learned cone angle bounds. On the right, the trajectories in solid deep/light blue correspond to reproductions w/o CCBF-QP, while the dashed black trajectories represent the reference trajectories.

constraints, we calculated the normalized average constraint violation (NACV) for all axes, i.e.

$$\text{NACV} = \frac{1}{3T}\Sigma_{t=0}^{T}\Sigma_{i=1}^{3}\max\{0, (\theta_i^{uc/c}(t) - \theta_i(t))/\theta_i(t)\},$$

where $\theta_i^{uc/c}$ refers to the angles between the actual and the reference cone axes for unconstrained/constrained runs. We obtained NACV = 2.033 ± 0.844 for unconstrained executions, and, as expected, NACV = 0 (no violations) for executions subject to constraints.

B. Robot Experiment

In this experiment, we focused on learning and executing a cutting skill in an assisted teleoperation scenario. Cutting tasks demand continuous orientation adjustments along the trajectory with particular significance placed on the tilt angle of the blades, since it determines incision shapes as illustrated in Fig. 1. In applications such as robotic surgery for tissue resection, operators may require precise control over the incision shape by adjusting the blade tilt during skill execution. Meanwhile, it's crucial to confine the tilt angle within a predefined range to ensure safety. Thus, we imposed constraints on the tilt angle of the blade, which coincides with the z-axis of the EE frame.

Our hardware setup involved a 7 degrees of freedom manipulator *Franka Research 3* for skill execution and a haptic device *lambda.7* for teleoperation. The manipulator was controlled by a Cartesian impedance controller with a translational stiffness matrix of $\text{diag}([1500, 1500, 1500])$ N/m and a rotational stiffness matrix of $\text{diag}([75, 75, 75])$ Nm/rad. The damping matrices were set for critical damping.

To capture the cutting skill, We recorded five kinesthetic demonstrations. The hyperparameters of GMM and LWR remained the same as in the simulation. The minimum threshold of cone angle was set to 0.02 rad.

In the performance evaluation, we invited six users, each performing the test twice. The users introduced different inclinations of the cutting edge, potentially or deliberately exceeding the learned conic constraints around the z-axis of the EE frame. Fig. 7 shows the temporal evolution of cone angles and the rotational trajectories of the EE frame in two independent runs. We observed that the executions with CBF consistently fulfilled the learned constraints (the NACV of the z-axis is $\text{NACV}_z = 0$), while unconstrained ones deviated far from the reference trajectories under excessive operator inputs ($\text{NACV}_z = 2.218 \pm 1.025$).

VI. CONCLUSIONS

In summary, we presented an innovative method for the safe execution of learned rotation skills by leveraging the combination of (*i*) a stable DS on $SO(3)$ and (*ii*) CCBFs that guarantee satisfaction of (*iii*) time-varying conic constraints extracted from the variability of demonstrations. We showed in simulation that our approach ensured that the learned DS converged to the goal, evolved closely to the reference trajectory, and always stayed within the region defined by the conic constraints. Furthermore, we have showcased the practical applicability of our approach in an assisted teleoperation scenario for cutting skills. In our future work, we intend to enhance the constraint learning method, currently restricted to trajectories whose distance to the target monotonically decreases. Moreover, we aim to improve the approach's usability by encompassing both translational and rotational motions based on coupled DSs [34].

REFERENCES

[1] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Survey: Robot programming by demonstration," Springrer, Tech. Rep., 2008.

[2] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[3] N. Figueroa and A. Billard, "Locally active globally stable dynamical systems: Theory, learning, and experiments," *The International Journal of Robotics Research*, vol. 41, no. 3, pp. 312–347, 2022.

[4] N. B. Figueroa Fernandez and A. Billard, "A physically-consistent bayesian non-parametric mixture model for dynamical system learning," *Proceedings of Machine Learning Research*, 2018.

[5] T. Ibuki, S. Wilson, A. D. Ames, and M. Egerstedt, "Distributed collision-free motion coordination on a sphere: A conic control barrier function approach," *IEEE Control Systems Letters*, vol. 4, no. 4, pp. 976–981, 2020.

[6] 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.

[7] M. Saveriano, F. J. Abu-Dakka, A. Kramberger, and L. Peternel, "Dynamic movement primitives in robotics: A tutorial survey," *The International Journal of Robotics Research*, 2023.

[8] M. J. Zeestraten, I. Havoutis, J. Silvério, 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.

[9] F. J. Abu-Dakka, M. Saveriano, and V. Kyrki, "A unified formulation of geometry-aware dynamic movement primitives," *arXiv preprint arXiv:2203.03374*, 2022.

[10] M. Saveriano, F. J. Abu-Dakka, and V. Kyrki, "Learning stable robotic skills on riemannian manifolds," *Robotics and Autonomous Systems*, p. 104510, 2023.

[11] A. L. P. Ureche, K. Umezawa, Y. Nakamura, and A. Billard, "Task parameterization using continuous constraints extracted from human demonstrations," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1458–1471, 2015.

[12] M. Menner, P. Worsnop, and M. N. Zeilinger, "Constrained inverse optimal control with application to a human manipulation task," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 2, pp. 826–834, 2019.

[13] M. Saveriano and D. Lee, "Learning barrier functions for constrained motion planning with dynamical systems," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 112–119.

[14] M. Davoodi, A. Iqbal, J. M. Cloud, W. J. Beksi, and N. R. Gans, "Probabilistic movement primitive control via control barrier functions," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2021, pp. 697–703.

[15] G. Sutanto, I. R. Fernández, P. Englert, R. K. Ramachandran, and G. Sukhatme, "Learning equality constraints for motion planning on manifolds," in *Conference on Robot Learning*. PMLR, 2021, pp. 2292–2305.

[16] N. Mehr, R. Horowitz, and A. D. Dragan, "Inferring and assisting with constraints in shared autonomy," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 6689–6696.

[17] C. Pérez-D'Arpino and J. A. Shah, "C-learn: Learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 4058–4065.

[18] G. Chou, D. Berenson, and N. Ozay, "Learning constraints from demonstrations with grid and parametric representations," *The International Journal of Robotics Research*, vol. 40, no. 10-11, pp. 1255–1283, 2021.

[19] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European control conference (ECC)*. IEEE, 2019, pp. 3420–3431.

[20] M. Igarashi, I. Tezuka, and H. Nakamura, "Time-varying control barrier function and its application to environment-adaptive human assist control," *IFAC Symposium on Nonlinear Control Systems*, vol. 52, no. 16, pp. 735–740, 2019.

[21] G. Wu and K. Sreenath, "Safety-critical geometric control for systems on manifolds subject to time-varying constraints," *IEEE Transactions on Automatic Control (TAC), in review*, 2016.

[22] X. Tan and D. V. Dimarogonas, "Construction of control barrier function and C^2 reference trajectory for constrained attitude maneuvers," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 3329–3334.

[23] L. B. Rosenberg, "Virtual fixtures: Perceptual tools for telerobotic manipulation," in *Proceedings of IEEE virtual reality annual international symposium*. Ieee, 1993, pp. 76–82.

[24] M. Ewerton, O. Arenz, and J. Peters, "Assisted teleoperation in changing environments with a mixture of virtual guides," *Advanced Robotics*, vol. 34, no. 18, pp. 1157–1170, 2020.

[25] A. D. Dragan and S. S. Srinivasa, "A policy-blending formalism for shared control," *The International Journal of Robotics Research*, vol. 32, no. 7, pp. 790–805, 2013.

[26] S. Javdani, H. Admoni, S. Pellegrinelli, S. S. Srinivasa, and J. A. Bagnell, "Shared autonomy via hindsight optimization for teleoperation and teaming," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 717–742, 2018.

[27] I. Havoutis and S. Calinon, "Learning assistive teleoperation behaviors from demonstration," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2016, pp. 258–263.

[28] ——, "Supervisory teleoperation with online learning and optimal control," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1534–1540.

[29] C. E. Mower, J. Moura, and S. Vijayakumar, "Skill-based shared control." in *Robotics: Science and Systems*, 2021, pp. 1–10.

[30] F. Bullo and A. D. Lewis, *Geometric control of mechanical systems: modeling, analysis, and design for simple mechanical control systems*. Springer, 2019, vol. 49.

[31] L. Lindemann and D. V. Dimarogonas, "Control barrier functions for signal temporal logic tasks," *IEEE control systems letters*, vol. 3, no. 1, pp. 96–101, 2018.

[32] R. M. Murray, Z. Li, and S. S. Sastry, *A mathematical introduction to robotic manipulation*. CRC press, 2017.

[33] S. Berkane and A. Tayebi, "Construction of synergistic potential functions on so(3) with application to velocity-free hybrid attitude stabilization," *IEEE Transactions on Automatic Control*, vol. 62, no. 1, pp. 495–501, 2017.

[34] A. Shukla and A. Billard, "Coupled dynamical system based arm–hand grasping model for learning fast adaptation strategies," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 424–440, 2012.