Probabilistic Adaptive Control for Robust Behavior Imitation

Julius Jankowski, Hakan Girgin and Sylvain Calinon

***Abstract*—In the context of learning from demonstration (LfD), trajectory policy representations such as probabilistic movement primitives (ProMPs) allow for rich modeling of demonstrated skills. To reproduce a learned skill with a real robot, a feedback controller is required to cope with perturbations and to react to dynamic changes in the environment. In this paper, we propose a generalized probabilistic control approach that merges the probabilistic modeling of the demonstrated movements and the feedback control action for reproducing the demonstrated behavior. We show that our controller can be easily employed, outperforming both original controller and a controller with constant feedback gains. Furthermore, we show that the proposed approach is able to solve dynamically changing tasks by modeling the demonstrated behavior as Gaussian mixtures and by introducing context variables. We demonstrate the capability of the approach with experiments in simulation and by teaching a 7-axis Franka Emika Panda robot to drop a ball into a moving box with only few demonstrations.**

***Index Terms*—Imitation Learning; Machine Learning for Robot Control; Robust/Adaptive Control**

I. Introduction

TRADITIONAL methods in robot programming require significant engineering expertise. Learning from demonstration (LfD) is a research field that overcomes these difficulties by teaching adaptive behaviors to robots via demonstrations. As a broad view, LfD approaches use demonstrations to learn the underlying control objectives (known as inverse optimal control (IOC)) or directly learn the policies that imitate the demonstrated behaviors. Although IOC methods are, in theory, able to achieve a more generalized reproduction of the demonstrations, in practice, they are most often limited by the reparametrization of the objective function and the time required for learning these parameters. For this reason, in LfD, learning the control/trajectory policies is often investigated as they are more flexible and easy to learn [1].

The objective of policy learning algorithms is to capture important characteristics of the demonstrations such as variations and multimodality in order to robustly reproduce the learned skill in novel environments. For that, demonstrated skills can be represented as a probability distribution in the state space [2], [3] and additionally in task-parameterized state spaces [4].

Manuscript received: October 15, 2020; Revised December 07, 2020; Accepted January 31, 2021. This paper was recommended for publication by Editor D. Kulic upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the CODIMAN project (Swiss National Science Foundation), and by the COLLABORATE project (https://collaborate-project.eu/, funded by the EU within H2020-DT-FOF-02-2018 under grant agreement 820767).

The authors are with the Idiap Research Institute, Martigny, Switzerland and with EPFL, Lausanne, Switzerland. julius.jankowski@idiap.ch

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

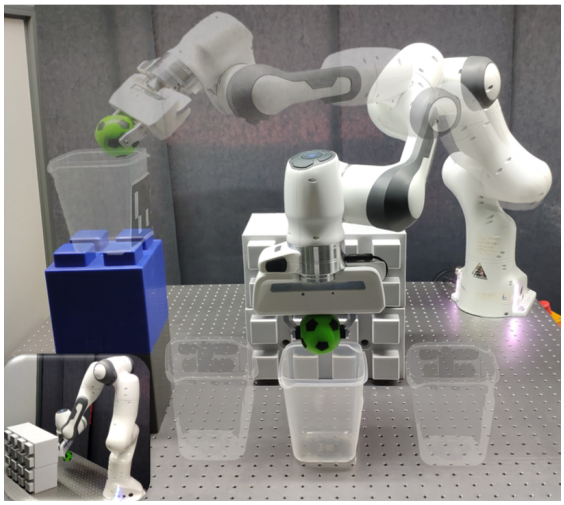

Fig. 1. The considered task for the robot is to drop a ball into a moving box using LfD. The robot starts from the pose illustrated by the picture in the lower-left corner.

To capture correlations along the time horizon, [5] proposes to represent skills as a distribution in the parameter space of parameterized trajectories. To avoid the necessity of a known dynamic model of the robot, [6] proposes to additionally include the control action that has been applied during the demonstration. In [7], demonstrated skills are modeled as the result of linear second-order dynamics. These approaches have later been extended to multimodality [8]–[11] and the realtime adaptation to context variables [12], [13].

In this paper, we propose a probabilistic control approach based on trajectory distributions. The proposed approach is tested with ProMPs, but it also extends to other forms of trajectory distributions if there is a linear relation between the corresponding latent space and the trajectory space, see [14] for a review. We show that the probabilistic control formulation encodes a compromise between a robust trajectory tracking controller and an imitation controller that adapts its gains to the demonstrated variations of the movement. Our contributions are three-fold:

- We propose a controller that outperforms the controller presented in [5] in terms of robustness, which is necessary for practical applications, while preserving the ability to reproduce the demonstrated variations;
- We show that the approach can readily be extended to exploit multiple modes in realtime;
- We develop a control formulation that can feed back general context variables in realtime.

The structure of the paper is as follows. In Section II, we review state-of-the-art work on controller design in ProMPs.

In Section III, we present our control approach by deriving control policies for the single-mode case III-B, the multimodal case III-C, and for context adaptation III-D. In Section IV, we compare the results to the controller in [5] and evaluate the practical use of the presented approach in an experiment with a 7-axis Franka Emika Panda robot. We conclude the paper and discuss further work in Section V.

II. Related Work

Paraschos *et al.* introduced the use of ProMPs to model demonstrated trajectories in a compact and probabilistic way [5]. The key characteristic of ProMPs is the parameterized representation of trajectories by separating time-dependent features $\boldsymbol{\Phi}(t)$ and parameters $\boldsymbol{w}$. For a given time t, the corresponding position $\boldsymbol{q}(t)$ and velocity $\dot{\boldsymbol{q}}(t)$ is encoded as

$$\boldsymbol{q}(t) = \boldsymbol{\Phi}(t)\boldsymbol{w}, \quad \dot{\boldsymbol{q}}(t) = \dot{\boldsymbol{\Phi}}(t)\boldsymbol{w}. \tag{1}$$

To track the trajectory distribution encoded by the learned probabilistic model $p(\boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w)$, Paraschos *et al.* derive a time-varying feedback controller by matching the rolled-out closed-loop system dynamics with the robot state distribution of the ProMP model. Therefore, the system dynamics are linearized, while assuming perfect knowledge of the system dynamics. Thus, unmodeled perturbations may lead to strong deviations from the expected behavior. Furthermore, they do not show how their controller can be extended to the multimodal case or to dynamically changing context variables. In the following, we refer to their controller as *original controller*, which is used as a baseline to benchmark our proposed controller.

More recently, Paraschos *et al.* proposed to learn a joint distribution of the motion trajectory and the recorded control action along the trajectory [6]. The controller is derived by conditioning the control action distribution on the current robot position and velocity to obtain a control policy distribution. They propose to stabilize the controlled system by adding linear feedback terms to let the robot converge back to the mean of the trajectory distribution only if the robot is 'far' from the demonstrated region in the state space. This approach does not require the knowledge of the system dynamics. However, it is limited to the collection of demonstrations only through teleoperation, e.g. by capturing the motion of a human demonstrator and executing the motion on the robot in realtime, to be able to record the control actions.

Ewerton *et al.* [8] propose to model multiple modes for a single task or multiple tasks by using a mixture of ProMP models. They propose to select a particular mode in advance of the execution based on the context of the task. This strategy does not allow the robot to adapt to changes in the environment and to choose another mode on-the-fly.

Ewerton *et al.* [12] tackle the problem of dynamic environment states, e.g. moving obstacle, by learning the parameters of a Gaussian process that outputs a ProMP distribution in realtime based on the current state of the environment. The underlying robot controller tracks the mean of the resulting ProMP distribution with constant feedback gains. To achieve a good extrapolation of ProMP distribution parameters with the Gaussian process, they pre-solve the task in simulation by designing objective functions and optimizing a trajectory for several random environment states. Although they show a successful realtime adaptation to dynamic environments, their approach requires the design of an objective function that encodes the solution of the task.

III. Control Formulation and Analysis

The rigid-body dynamics of a robot manipulator with n degrees of freedom is described as

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u} + \boldsymbol{\tau}_{\text{ext}}, \tag{2}$$

where $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ are the joint positions, velocities and accelerations, respectively. The inertia matrix $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$, the Coriolis and centrifugal matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{n\times n}$ and the gravitational term $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^n$ are the state-dependent parameters of the dynamics model, with state represented by $\boldsymbol{y}{=}[\boldsymbol{q}^\top, \dot{\boldsymbol{q}}^\top]^\top$. The torque-control action $\boldsymbol{u} \in \mathbb{R}^n$ and the external torques $\boldsymbol{\tau}_{\text{ext}} \in \mathbb{R}^n$ acting on the robot form the input to the intrinsic robot dynamics.

The objective of this section is to find a time-dependent feedback controller $\boldsymbol{u}$, that enables the closed-loop system to robustly and reactively imitate demonstrated behavior at a kinematic level. Each demonstration k contains the robot's position and velocity trajectories $\{\boldsymbol{q}_t^k, \dot{\boldsymbol{q}}_t^k\}_{t=1}^T$ with a time horizon T, and some optional context variables $\boldsymbol{s}_k$ describing the information that affects the whole trajectory, such as environment properties (e.g. size of objects). We denote the demonstration set as $\boldsymbol{\mathcal{D}} = \left\{\{\boldsymbol{q}_t^k, \dot{\boldsymbol{q}}_t^k\}_{t=1}^T, \boldsymbol{s}_k\right\}_{k=1}^K$ where K denotes the number of demonstrations.

Oftentimes, demonstrated tasks can be captured more efficiently by recording the Cartesian space trajectories of the end-effector of a manipulator instead of the joint space trajectories. This requires the formulation of the controller in the corresponding space. Since the mathematical structure of the system dynamics in the Cartesian space is similar to the joint space formulation, the following derivations and analyses are also applicable to the Cartesian case. We provide the Cartesian space dynamics in the Appendix.

We denote $\boldsymbol{w}$ as the weight vectors whose elements can be interpreted as activations of time-dependent trajectory features $\boldsymbol{\Phi}(t)$. These trajectory features are often designed as radial basis functions with centers equally distributed in time and bandwidth selected manually to capture the demonstration data. Other forms of basis functions such as Bernstein polynomials or Fourier series can also be used within the ProMP framework [15], [16].

In this paper, we do not set focus on the algorithms that learn a probability distribution $p(\boldsymbol{w}|\boldsymbol{\mathcal{D}})$ for the weight vectors $\boldsymbol{w}$ from the demonstration data $\boldsymbol{\mathcal{D}}$. Instead, we refer to the previous work on ProMPs [5], [17] where these algorithms have been presented.

Next, we derive the proposed controller in Section III-A, that we present for a single mode (Section III-B), for multiple modes (Section III-C), and for multiple modes with a context variable (Section III-D). Each case can be considered a more general representation of the previous ones, with the context-based multimodal case representing the main result of this paper.

In all sections, we compare in simulated experiments the proposed controller with the original controller and a controller with constant feedback gains. Therefore, we measure the *Effort* during a rollout as $C_u = \int \boldsymbol{u}^\top \boldsymbol{u}\, dt$ and we measure the *Tracking Error* as $C_e = -\int \log(p(\boldsymbol{q}, \dot{\boldsymbol{q}}|\boldsymbol{\mathcal{D}}, \boldsymbol{s})dt$.

A. Formulation of the Controller

As an underlying control structure, we propose to use a compliant controller

$$\boldsymbol{u} = -\boldsymbol{K}(\boldsymbol{q} - \boldsymbol{q}_d) - \boldsymbol{D}(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d) + \boldsymbol{M}\ddot{\boldsymbol{q}}_d + \boldsymbol{C}\dot{\boldsymbol{q}}_d + \boldsymbol{g}, \quad (3)$$

that enables a robot with the dynamics in (2) to track a reference trajectory $\{\boldsymbol{q}_d{=}\boldsymbol{\Phi}(t)\boldsymbol{w},\, \dot{\boldsymbol{q}}_d{=}\dot{\boldsymbol{\Phi}}(t)\boldsymbol{w},\, \ddot{\boldsymbol{q}}_d{=}\ddot{\boldsymbol{\Phi}}(t)\boldsymbol{w}\}_{t=1}^{T}$ with the user-defined positive definite feedback gains $\boldsymbol{K}, \boldsymbol{D} \in \mathbb{R}^{n\times n}$. Note that for a stationary $\boldsymbol{w}$, the closed-loop system controlled by this controller is known to be asymptotically stable as pointed out in III-B1. The resulting closed-loop dynamics are linear in the trajectory weights $\boldsymbol{w}$, which can also be interpreted as a latent auxiliary control input. In contrast to the original control formulation, this avoids the necessity of linearizing the passive system dynamics in (2) and introduces a stabilizing feedback structure.

As in [5], we model the weight vector $\boldsymbol{w}$ as a random variable that correlates with the demonstrations $\boldsymbol{\mathcal{D}}$ and the current robot state $\boldsymbol{y}$. Thus, we find a probabilistic imitation controller by marginalizing over the weight vectors with

$$p(\boldsymbol{u}|\boldsymbol{y}, \boldsymbol{\mathcal{D}}) = \int_{\boldsymbol{w}} p(\boldsymbol{u}|\boldsymbol{y}, \boldsymbol{w}) p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\mathcal{D}}) d\boldsymbol{w}. \quad (4)$$

Paraschos *et al.* derive their controller in a similar fashion [6], however, we propose to exploit the compliant control structure in (3) instead of learning a joint distribution of robot states and control actions. The probability distribution $p(\boldsymbol{u}|\boldsymbol{y}, \boldsymbol{w})$ focuses all the probability mass at the deterministic controller given in (3). Hence, the probability distribution can be written as a Dirac function, such that

$$\begin{aligned} p(\boldsymbol{u}|\boldsymbol{y}, \boldsymbol{w}) &= \delta(\boldsymbol{A}\boldsymbol{w} + \boldsymbol{b}), \quad (5)\\ \text{with} \quad \boldsymbol{A} &= \boldsymbol{K}\boldsymbol{\Phi} + (\boldsymbol{C} + \boldsymbol{D})\dot{\boldsymbol{\Phi}} + \boldsymbol{M}\ddot{\boldsymbol{\Phi}},\\ \boldsymbol{b} &= -\boldsymbol{K}\boldsymbol{q} - \boldsymbol{D}\dot{\boldsymbol{q}} + \boldsymbol{g}. \end{aligned}$$

It can be seen that the nonlinear feedback controller is linear w.r.t. the weight vector $\boldsymbol{w}$, which results in tractable control distributions.

The conditional distribution $p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\mathcal{D}})$ of the weight vectors can be reformulated by Bayes' theorem

$$p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\mathcal{D}}) \propto p(\boldsymbol{y}|\boldsymbol{w}) p(\boldsymbol{w}|\boldsymbol{\mathcal{D}}), \quad (6)$$

such that the learned ProMP model $p(\boldsymbol{w}|\boldsymbol{\mathcal{D}})$ evolves from the formulation of a probabilistic imitation controller. We complete the probabilistic model in (6) by letting the robot state $\boldsymbol{y}$ be correlated with the current weight vector by using a linear Gaussian model $p(\boldsymbol{y}|\boldsymbol{w}){=}\mathcal{N}(\boldsymbol{\Phi}\boldsymbol{w}, \boldsymbol{\Sigma}_{\boldsymbol{q}})\mathcal{N}(\dot{\boldsymbol{\Phi}}\boldsymbol{w}, \boldsymbol{\Sigma}_{\dot{\boldsymbol{q}}})$. Here, the covariance matrices $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ and $\boldsymbol{\Sigma}_{\dot{\boldsymbol{q}}}$ can be understood as a tracking tolerance for the controller w.r.t. a given weight vector $\boldsymbol{w}$. Within the following section, we show that the design of these tolerances plays an important role in finding a compromise between a robust controller and a controller with time-varying feedback gains that imitates the variations of the motion.

B. Single Mode

For simple tasks, e.g. reaching tasks in non-cluttered environments, it is sufficient to learn a single Gaussian distribution of weight vectors $\boldsymbol{w}$ to model the demonstrated movement. In these cases, as also investigated in the original work [5], the learned distribution of weight vectors is given by $p(\boldsymbol{w}|\boldsymbol{\mathcal{D}}) = \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{w}}, \boldsymbol{\Sigma}_{\boldsymbol{w}})$. As a result, the distribution in (6) can be computed closed-form as

$$p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\mathcal{D}}) = \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}}, \boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{y}}), \quad (7)$$

where the conditional mean $\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}}$ depends on time and the robot state.

Since the result is a Gaussian distribution, the solution of the integral in (4) becomes analytically tractable. Hence, we obtain a state and time-dependent Gaussian distribution of feedback control actions that are conditioned on the demonstrations, namely

$$p(\boldsymbol{u}|\boldsymbol{y}, \boldsymbol{\mathcal{D}}) = \mathcal{N}(\boldsymbol{A}\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}} + \boldsymbol{b},\ \boldsymbol{A}\boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{y}}\boldsymbol{A}^\top), \quad (8)$$

with $\boldsymbol{A}$ and $\boldsymbol{b}$ given by (5). Obtaining a Gaussian distribution of control actions can be exploited by a *product of experts* scheme in order to blend multiple complementary probabilistic controllers, as shown in [18], [19]. However, in this paper, we focus on the imitation performance when using the most likely control action that is given by the mean of the control distribution

$$\begin{aligned} \boldsymbol{\mu}_{\boldsymbol{u}} = -\tilde{\boldsymbol{K}}(\boldsymbol{q} - \boldsymbol{\Phi}\boldsymbol{\mu}_{\boldsymbol{w}}) - \tilde{\boldsymbol{D}}(\dot{\boldsymbol{q}} - \dot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}}) +\\ \boldsymbol{M}\ddot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}} + \boldsymbol{C}\dot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}} + \boldsymbol{g}. \quad (9) \end{aligned}$$

The feedback gain matrices are given by

$$\begin{aligned} \tilde{\boldsymbol{K}} &= \boldsymbol{K} - \boldsymbol{A}\boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{y}}\boldsymbol{\Phi}^\top\boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1},\\ \tilde{\boldsymbol{D}} &= \boldsymbol{D} - \boldsymbol{A}\boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{y}}\dot{\boldsymbol{\Phi}}^\top\boldsymbol{\Sigma}_{\dot{\boldsymbol{q}}}^{-1}. \end{aligned} \quad (10)$$

This shows that the resulting controller has time-varying feedback gains that depend on the variations of the demonstrations. To understand the behavior of the closed-loop system controlled by the proposed controller, we analyze two limit cases of designing the tracking tolerances $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ and $\boldsymbol{\Sigma}_{\dot{\boldsymbol{q}}}$. The first case in Section III-B1 shows that setting infinitely large tolerances corresponds to a provably asymptotically stable controller that tracks the mean of the demonstrated trajectories. The second case in Section III-B2 shows that setting close to zero tolerances corresponds to a reproduction of the demonstrated variations of the trajectory distribution in the absence of perturbations.

1) Infinitely Large Tolerance - Asymptotic Stability: In the limit case of infinitely high eigenvalues for the tracking tolerances $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ and $\boldsymbol{\Sigma}_{\dot{\boldsymbol{q}}}$, the feedback gains in (10) become equal to $\boldsymbol{K}$ and $\boldsymbol{D}$, respectively. As a result, the controller tracks the mean of the demonstrated movements with constant

feedback gains. It can be shown that the corresponding closed-loop system is asymptotically stable by proving that the candidate Lyapunov function

$$S = \tfrac{1}{2}(\boldsymbol{q} - \boldsymbol{\Phi}\boldsymbol{\mu}_{\boldsymbol{w}})^\top \boldsymbol{K}(\boldsymbol{q} - \boldsymbol{\Phi}\boldsymbol{\mu}_{\boldsymbol{w}}) + \tfrac{1}{2}(\dot{\boldsymbol{q}} - \dot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}})^\top \boldsymbol{M}(\dot{\boldsymbol{q}} - \dot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}}) \quad (11)$$

satisfies the necessary conditions $\{S \geq 0,\ \dot{S} \leq 0,\ \forall \boldsymbol{y}\}$. However, it can be seen that the resulting controller does not depend on the learned weight vector covariance $\boldsymbol{\Sigma}_{\boldsymbol{w}}$, which encodes the variations of the demonstrated behavior. Thus, this controller is neither able to reproduce nor exploit these variations.

2) Infinitely Small Tolerance - Matching the Distribution: In the other limit case of close to zero eigenvalues for the tracking tolerances $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ and $\boldsymbol{\Sigma}_{\dot{\boldsymbol{q}}}$, we observe that the proposed controller becomes similar to the controller that was proposed in [5]. Note that this analysis only holds for linear system dynamics, e.g. a virtual point mass, where $\boldsymbol{M}$ does not depend on the system state, $\boldsymbol{C}{=}\boldsymbol{0}$ and $\boldsymbol{g}{=}\boldsymbol{0}$. For this case, the proposed controller simplifies to

$$\boldsymbol{\mu}_{\boldsymbol{u}} = \boldsymbol{M}\ddot{\boldsymbol{\Phi}}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top(\boldsymbol{\Psi}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top)^{-1}(\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{\mu}_{\boldsymbol{w}}) + \boldsymbol{M}\ddot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}}, \quad (12)$$

where $\boldsymbol{\Psi}{=}[\boldsymbol{\Phi}^\top, \dot{\boldsymbol{\Phi}}^\top]^\top$ is the stacked feature matrix. It can be observed that the proportional and differential feedback terms including $\boldsymbol{K}$ and $\boldsymbol{D}$ vanished from the proposed controller because of $\boldsymbol{\Phi}\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}}{=}\boldsymbol{q}$ and $\dot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}}{=}\dot{\boldsymbol{q}}$ in the case of infinitely small tolerance eigenvalues. The vanishing of those feedback terms is expected to reduce the robustness of the closed-loop system against perturbations. The derivation of this result is given in the appendix.

The controller in [5] has been derived by matching the closed-loop system dynamics with the learned evolution of the demonstrated movements. For the case of a virtual point mass with a force-control interface, it is given by

$$\boldsymbol{\mu}_{\boldsymbol{u},\text{orig}} = \boldsymbol{M}\ddot{\boldsymbol{\Phi}}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top(\boldsymbol{\Psi}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top)^{-1}(\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{\mu}_{\boldsymbol{w}}) + \boldsymbol{M}\ddot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}} - \frac{1}{2}[\boldsymbol{0}, \boldsymbol{M}]\boldsymbol{\Sigma}_s(\boldsymbol{\Psi}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top)^{-1}(\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{\mu}_{\boldsymbol{w}}), \quad (13)$$

where the first two terms are equal to the proposed controller in (12). We assume that the eigenvalues of the matrix $\boldsymbol{\Sigma}_s$, which is derived from the cross-correlation between two consecutive time steps, converge to zero as the duration of the time step tends towards zero. Thus, we assume that the third term is negligible for standard control frequencies of 1 kHz. Consequently, for infinitely small tolerance eigenvalues, we find $\boldsymbol{\mu}_{\boldsymbol{u}} \approx \boldsymbol{\mu}_{\boldsymbol{u},\text{orig}}$.

The analysis of the two limit cases shows that the proposed controller encodes a compromise between a robust tracking of the demonstrated mean trajectory and a reproduction of the demonstrated variations. This compromise is designed by selecting the tracking tolerances $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ and $\boldsymbol{\Sigma}_{\dot{\boldsymbol{q}}}$.

We validate our analysis and the underlying assumptions in Fig. 2 by simulating the proposed controller, the original formulation and a controller with constant feedback gains as in III-B1 (*Stiff*) in a single-mode drawing task. The left plot shows that the proposed controller with close-to-zero tolerances and the original controller result in a similar behavior

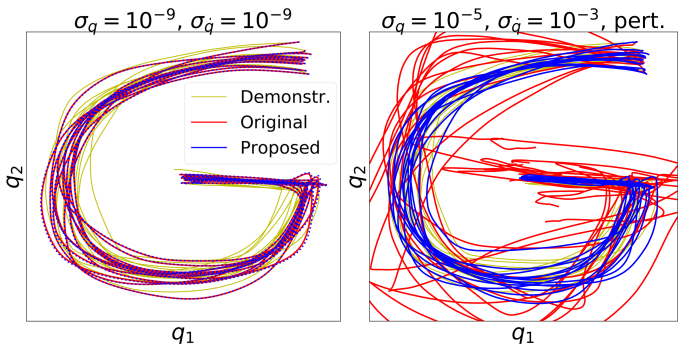

Fig. 2. Drawing task for a 2D point mass. The parameters of the proposed controller are set to $\boldsymbol{\Sigma}_{\boldsymbol{q}}{=}\sigma_{\boldsymbol{q}}\boldsymbol{I}$, $\boldsymbol{\Sigma}_{\dot{\boldsymbol{q}}}{=}\sigma_{\dot{\boldsymbol{q}}}\boldsymbol{I}$, $\boldsymbol{K}{=}500\boldsymbol{I}$ and $\boldsymbol{D}{=}3\sqrt{500}\boldsymbol{I}$. The original controller and the proposed controller produce the same behavior if the tolerance of tracking the probabilistic model is set close to zero (left plot). For the tolerances $\sigma_{\boldsymbol{q}}{=}10^{-5}$ and $\sigma_{\dot{\boldsymbol{q}}}{=}10^{-3}$, the proposed controller adds stabilizing feedback terms. This results in a probabilistic feedback control that is robust to perturbations (right plot).

TABLE I
METRIC-BASED COMPARISON FOR THE DRAWING TASK (FIG. 2).

Method and Scenario	Effort	Tracking Error	KL-Div.
Original, no pert.	5.5	0.91	85.0
Proposed (a), no pert.	5.5	0.91	85.1
Proposed (b), no pert.[1]	4.35	0.75	197.6
Stiff, no pert.[1]	3.52	0.72	770.2
Original, perturbed	17.9	6.39	40.4
Proposed (a), perturbed[1]	17.9	6.39	40.4
Proposed (b), perturbed	5.99	1.25	40.2
Stiff, perturbed[1]	4.23	1.09	270.5

as derived in III-B2. The plot on the right side illustrates the trajectories when there is a random force $\boldsymbol{\tau}_{\text{ext}} \sim \mathcal{N}(\boldsymbol{0}, 10\boldsymbol{I})$ perturbing the system and the tolerances are set to $\sigma_{\boldsymbol{q}}{=}10^{-5}$ and $\sigma_{\dot{\boldsymbol{q}}}{=}10^{-3}$. The proposed controller shows higher robustness to the perturbations than the original controller. Table I shows the evaluated controller metrics for the perturbed and unperturbed case. The proposed controller is evaluated using two instances with tolerances set to $\sigma_{\boldsymbol{q}}{=}10^{-9}$, $\sigma_{\dot{\boldsymbol{q}}}{=}10^{-9}$ (a) and $\sigma_{\boldsymbol{q}}{=}10^{-5}$, $\sigma_{\dot{\boldsymbol{q}}}{=}10^{-3}$ (b), respectively. In terms of the effort and the tracking error, the stiff controller shows the best performance for this single-mode tracking task. However, the evaluation of the Kullback-Leibler divergence between the demonstrations and each set of rollouts reflects that the stiff controller does not take the stochastic distribution of the demonstrations into account.

C. Multiple Modes

For more complicated tasks, it can be desirable to use a more expressive model for the distribution of trajectory weight vectors in order to capture multiple modes. One way to do so is to use a mixture of Gaussians that is a weighted sum of N Gaussian components

$$p(\boldsymbol{w}|\boldsymbol{\mathcal{D}}) = \sum_{n=1}^{N} \pi_n \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{w}}^n, \boldsymbol{\Sigma}_{\boldsymbol{w}}^n), \quad (14)$$

where π_n is the mixing coefficient.The conditional distribution of a Gaussian mixture model is itself a Gaussian mixture

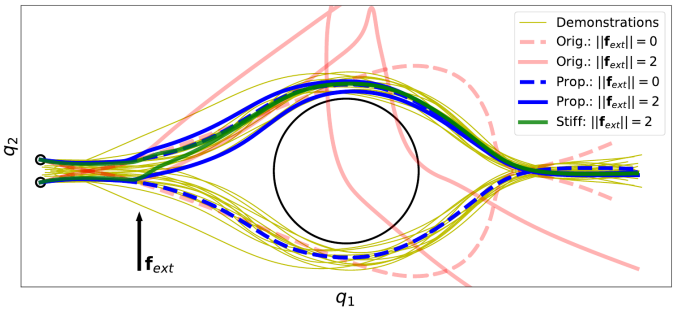

Fig. 3. Multiple-mode navigation task for a 2D point mass. The parameters of the proposed controller are set to $\boldsymbol{\Sigma_q}$=$10^{-5}\boldsymbol{I}$, $\boldsymbol{\Sigma_{\dot{q}}}$=$10^{-3}\boldsymbol{I}$, $\boldsymbol{K}$=$100\boldsymbol{I}$ and $\boldsymbol{D}$=$3\sqrt{100}\boldsymbol{I}$. The dashed trajectories show the behavior of the original controller (red) and the proposed controller (blue) without perturbations. The solid trajectories show the result of all controllers under the impact of an external force $\boldsymbol{f}_{\text{ext}}$ that pushes the point mass in positive q_2-direction for 0.1 seconds.

model given by

$$p(\boldsymbol{w}|\boldsymbol{y},\boldsymbol{\mathcal{D}}) = \sum_{n=1}^{N} \gamma_n(\boldsymbol{y}) \mathcal{N}(\boldsymbol{\mu}^n_{\boldsymbol{w}|\boldsymbol{y}}, \boldsymbol{\Sigma}^n_{\boldsymbol{w}|\boldsymbol{y}}), \tag{15}$$

where $\mathcal{N}(\boldsymbol{\mu}^n_{\boldsymbol{w}|\boldsymbol{y}}, \boldsymbol{\Sigma}^n_{\boldsymbol{w}|\boldsymbol{y}})$ is the conditional distribution of the n-th component and is computed in the same way as (7). The new mixing coefficients are given by

$$\gamma_n(\boldsymbol{y}) = \frac{\pi_n p_n(\boldsymbol{y}|\boldsymbol{\mathcal{D}})}{\sum_{l=1}^{N} \pi_l p_l(\boldsymbol{y}|\boldsymbol{\mathcal{D}})}, \tag{16}$$

$$\text{with} \quad p_n(\boldsymbol{y}|\boldsymbol{\mathcal{D}}) = \mathcal{N}(\boldsymbol{\Psi}\boldsymbol{\mu}^n_{\boldsymbol{w}}, \boldsymbol{\Sigma}_{\boldsymbol{y}} + \boldsymbol{\Psi}\boldsymbol{\Sigma}^n_{\boldsymbol{w}}\boldsymbol{\Psi}^\top), \tag{17}$$

which can be interpreted as the belief of the robot being in mode n. Here, the feature matrix is given by $\boldsymbol{\Psi}$=$[\boldsymbol{\Phi}^\top, \dot{\boldsymbol{\Phi}}^\top]^\top$, and the robot state covariance is given by $\boldsymbol{\Sigma_y}$=blockdiag$(\boldsymbol{\Sigma_q}, \boldsymbol{\Sigma_{\dot{q}}})$. This results in a mixture of Gaussian policies given by

$$\begin{aligned} p(\boldsymbol{u}|\boldsymbol{y},\boldsymbol{\mathcal{D}}) &= \sum_{n=1}^{N} \gamma_n(\boldsymbol{y}) \mathcal{N}(\boldsymbol{\mu}^n_{\boldsymbol{u}}, \boldsymbol{\Sigma}^n_{\boldsymbol{u}}), \\ \text{with} \quad \boldsymbol{\mu}^n_{\boldsymbol{u}} &= \boldsymbol{A}\boldsymbol{\mu}^n_{\boldsymbol{w}|\boldsymbol{y}} + \boldsymbol{b}, \\ \boldsymbol{\Sigma}^n_{\boldsymbol{u}} &= \boldsymbol{A}\boldsymbol{\Sigma}^n_{\boldsymbol{w}|\boldsymbol{y}}\boldsymbol{A}^\top. \end{aligned} \tag{18}$$

For the practical control of a robot, it is necessary to find the most likely control action. However, for a Gaussian mixture model, this corresponds to solving a non-convex optimization problem. Due to the time constraints of a realtime control loop, we propose to approximate the control distribution by finding the most likely component based on its mixture coefficient. Consequently, we use the mean control action of the selected component as the control input to the system such that $\boldsymbol{u}$=$\boldsymbol{\mu}^{n^*}_{\boldsymbol{u}}$ with $\gamma_{n^*}(\boldsymbol{y})$>$\gamma_n(\boldsymbol{y})$, $\forall n \neq n^*$.

From a theoretic perspective, the resulting closed-loop system corresponds to a hybrid system, where the function $\gamma_n(\boldsymbol{y})$ represents the guard which indicates the state-dependent switching between multiple closed-loop system dynamics that are all equivalent to the single-mode case that has been discussed in Section III-B.

Figure 3 shows the results for a navigation task with two modes that represent possible paths to avoid the collision with the obstacle (black circle). The synthetic demonstrations of the two modes have been separated in advance and the Gaussian components have been computed individually with π_1=π_2=0.5. For the implementation of the original controller and the stiff controller, we use the mixture coefficient computation of our proposed controller, given in (16), to find the most likely mode and apply the corresponding control command to the point mass. The dashed signals show the resulting trajectories in the absence of disturbances when started from the initial positions indicated by the small black circles. The solid signals show the resulting trajectories when there is a vertical force $\boldsymbol{f}_{\text{ext}}$ perturbing the point mass during 0.1 seconds. All controllers make use of the two modes in realtime by switching to the upper path after the point mass has been pushed in that direction. The original controller is not able to recover from the short-term perturbation that is caused by the external force and by the mode switching. The stiff controller and the proposed controller show a low tracking error also for the perturbed case. Table II shows that the proposed controller requires less effort than the stiff controller for the perturbed and unperturbed case.

[1]Not visualized in the figures for better readibility.

TABLE II
METRIC-BASED COMPARISON FOR MULTIPLE MODES (FIG. 3).

Method and Scenario	Effort	Tracking Error
Original, no pert.	0.27	0.75
Proposed, no pert.	0.14	0.55
Stiff, no pert.[1]	0.18	0.55
Original, pert., top	0.45	1.38
Proposed, pert., top	0.2	0.54
Stiff, pert., top	0.26	0.55
Original, pert., bottom	1.83	4.23
Proposed, pert., bottom	0.21	0.58
Stiff, pert., bottom	0.25	0.56

D. Feedback of the Context

To encode more general and adaptive skills, it is useful to introduce state-independent context variables that are supposed to affect the behavior of the robot. Context variables are also discussed in the original work of Paraschos *et al.* [5], however only considering the offline adaptation of a single ProMP distribution. Context variables can be used to encode information about the task, e.g. the position and size of the object to pick, but also information about the environment, e.g. the position and size of an obstacle. Since this information may change during the execution of a learned skill, it is desirable to incorporate the context as another random variable in the control policy to generate a reactive behavior. Thus, we reformulate the controller in (4) by augmenting the set of conditions by the context $\boldsymbol{s}$ with

$$p(\boldsymbol{u}|\boldsymbol{y},\boldsymbol{s},\boldsymbol{\mathcal{D}}) = \int_{\boldsymbol{w}} p(\boldsymbol{u}|\boldsymbol{w},\boldsymbol{y}) p(\boldsymbol{w}|\boldsymbol{y},\boldsymbol{s},\boldsymbol{\mathcal{D}}) d\boldsymbol{w}. \tag{19}$$

The conditional distribution of the weight vectors $\boldsymbol{w}$ is again given by Bayes' theorem

$$p(\boldsymbol{w}|\boldsymbol{y},\boldsymbol{s},\boldsymbol{\mathcal{D}}) = \frac{p(\boldsymbol{y}|\boldsymbol{w}) p(\boldsymbol{w},\boldsymbol{s}|\boldsymbol{\mathcal{D}})}{p(\boldsymbol{y},\boldsymbol{s}|\boldsymbol{\mathcal{D}})}. \tag{20}$$

Here, the joint probability distribution $p(\boldsymbol{w},\boldsymbol{s}|\boldsymbol{\mathcal{D}})$ can be learned by fitting a single Gaussian distribution as in [5] or

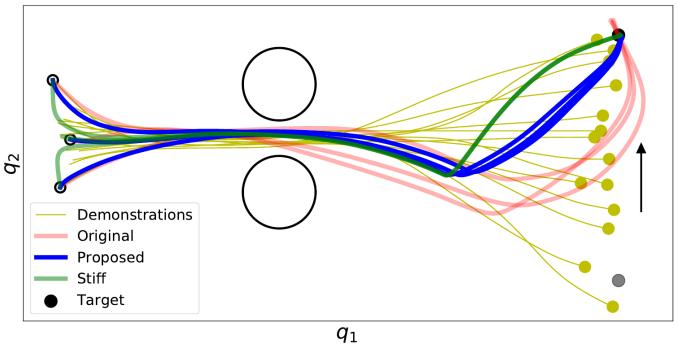

Fig. 4. Target reaching task for a 2D point mass initialized at three different positions depicted by the small black circles. The parameters of the proposed controller are set to $\boldsymbol{\Sigma_q}{=}10^{-5}\boldsymbol{I}$, $\boldsymbol{\Sigma_{\dot{q}}}{=}10^{-3}\boldsymbol{I}$, $\boldsymbol{K}{=}100\boldsymbol{I}$ and $\boldsymbol{D}{=}3\sqrt{100}\boldsymbol{I}$. The target, depicted by the filled black circle, jumps from the position indicated by the grey circle in the lower right corner to the final target in the upper right corner after 2.3 seconds. The duration of the motion is 4 seconds.

by fitting a Gaussian mixture model to the demonstrated data. For the sake of compactness, we directly consider the case of an arbitrary number N of Gaussian components representing the learned joint distribution

$$p(\boldsymbol{w},\boldsymbol{s}|\boldsymbol{\mathcal{D}}) = \sum_{n=1}^{N} \tilde{\pi}_n \mathcal{N}\left(\begin{pmatrix}\boldsymbol{\mu_w}\\ \boldsymbol{\mu_s}\end{pmatrix}^n, \begin{pmatrix}\boldsymbol{\Sigma_w} & \boldsymbol{\Sigma_{ws}}\\ \boldsymbol{\Sigma_{sw}} & \boldsymbol{\Sigma_s}\end{pmatrix}^n\right).$$

Analogously to the multimodality case in III-C, the result of the conditional distribution in (20) is given by a mixture of Gaussians

$$p(\boldsymbol{w}|\boldsymbol{y},\boldsymbol{s},\boldsymbol{\mathcal{D}}) = \sum_{n=1}^{N} \gamma_n(\boldsymbol{y},\boldsymbol{s})\mathcal{N}(\boldsymbol{\mu}^n_{\boldsymbol{w}|\boldsymbol{y},\boldsymbol{s}}, \boldsymbol{\Sigma}^n_{\boldsymbol{w}|\boldsymbol{y},\boldsymbol{s}}), \qquad (21)$$

where the mean of the n-th component additionally depends on the current context variable $\boldsymbol{s}$. The new component coefficient $\gamma_n(\boldsymbol{y},\boldsymbol{s})$ can be viewed as the belief of the robot being in mode n. It is given by

$$\gamma_n(\boldsymbol{y},\boldsymbol{s}) = \frac{\tilde{\pi}_n p_n(\boldsymbol{y}|\boldsymbol{s},\boldsymbol{\mathcal{D}}) p_n(\boldsymbol{s}|\boldsymbol{\mathcal{D}})}{\sum_{l=1}^{N} \tilde{\pi}_l p_l(\boldsymbol{y}|\boldsymbol{s},\boldsymbol{\mathcal{D}}) p_l(\boldsymbol{s}|\boldsymbol{\mathcal{D}})},$$
$$\text{with} \quad p_n(\boldsymbol{y}|\boldsymbol{s},\boldsymbol{\mathcal{D}}) = \mathcal{N}(\boldsymbol{\Psi}\boldsymbol{\mu}^n_{\boldsymbol{w}|\boldsymbol{s}}, \boldsymbol{\Sigma_y} + \boldsymbol{\Psi}\boldsymbol{\Sigma}^n_{\boldsymbol{w}|\boldsymbol{s}}\boldsymbol{\Psi}^\top),$$

where the marginalized context distribution is given by $p_n(\boldsymbol{s}|\boldsymbol{\mathcal{D}}){=}\mathcal{N}(\boldsymbol{\mu}^n_{\boldsymbol{s}}, \boldsymbol{\Sigma}^n_{\boldsymbol{s}})$. Similarly to Section III-C, we propose to approximate the Gaussian mixture distribution by the most likely component and to use the corresponding mean control action, such that $\boldsymbol{u} = \boldsymbol{A}\boldsymbol{\mu}^{n^*}_{\boldsymbol{w}|\boldsymbol{y},\boldsymbol{s}}{+}\boldsymbol{b}$ with $\gamma_{n^*}(\boldsymbol{y},\boldsymbol{s}) > \gamma_n(\boldsymbol{y},\boldsymbol{s})$, $\forall n \neq n^*$.

TABLE III
METRIC-BASED COMPARISON FOR A DYNAMIC CONTEXT (FIG. 4).

Method	Effort	Tracking Error
Original	0.25	8.93
Proposed	0.29	1.77
Stiff	0.93	1.38

Figure 4 illustrates the behavior of a 2D point mass for a single-mode goal reaching task where the target is jumping during the execution of the task. For the implementation of the original controller, we used its control formulation and replaced the stationary ProMP distribution parameters $\boldsymbol{\mu_w}$ and $\boldsymbol{\Sigma_w}$ by the context-conditional ProMP distribution parameters $\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{s}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{w}|\boldsymbol{s}}$. The *stiff* controller tracks the mean $\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{s}}$ of the conditional distribution with the constant feedback gains $\boldsymbol{K}$ and $\boldsymbol{D}$. In addition to the synthetic trajectory demonstrations, a target position, depicted by the yellow circle, has been recorded as a context variable $\boldsymbol{s}{=}\boldsymbol{q}_{target}$. During the simulation, the context changes by a jump from the gray filled circle to the black filled circle at $t{=}2.3$s. The full simulation duration is $t{=}4$s. The metric evaluation in Table III indicates that, in this experiment, a stiff controller requires an unnecessarily high effort while the original controller results in a high tracking error. The proposed controller combines the low effort of the original controller with the low tracking error of the stiff controller.

IV. EXPERIMENTAL VALIDATION

We conduct the experiments using a 7-axis Franka Emika Panda robot, by using the mathematical model of the system dynamics in (2). As this model does not include parasitic, nonlinear effects such as joint friction, these appear as inherent perturbations during the execution. This is the case in many robotic platforms and thus model-based controllers should be able to cope with these perturbations. The objective of the experiments is to show that our proposed controller can achieve this robustness while imitating reactively the demonstrated behavior and exploiting variations in realtime.

We consider the task of placing a ball inside a box in a cluttered environment. The task is fulfilled if the ball has been dropped into the box that is moving during the execution without colliding with the environment. Figure 1 shows the experimental setup, including the initial robot pose in the lower-left corner. The box position is detected by a stereo vision system and is used as a context variable. We provide 16 demonstrations of the robot end-effector trajectory using kinesthetic teaching. In Figure 5, each demonstrated trajectory is a solution to the task for a given context value (box position), depicted by yellow crosses. Note that the context values are fixed during each demonstration, such that tracking a moving box has not been demonstrated explicitly. We manually separate the demonstrations into two modes. The first mode encodes solutions for situations where the box is on the table with some variations, while the second mode encodes solutions when the box is placed on the blocks on the left with no variations.

We compute two individual ProMP models according to Section III-D by using the separated demonstrations. We then combine the individual models to obtain a Gaussian mixture model by computing the mixture coefficients according to the number of demonstrations provided for each mode. Each of the two components uses 12 radial basis functions as trajectory features. We implement the original formulation using an inverse dynamics approach as described in [5].

The parameters of the proposed controller are selected as $\boldsymbol{\Sigma_q}{=}10^{-5}\boldsymbol{I}$, $\boldsymbol{\Sigma_{\dot{q}}}{=}10^{-3}\boldsymbol{I}$, $\boldsymbol{K}{=}10^{3}\boldsymbol{I}$ and $\boldsymbol{D}{=}3\sqrt{10^3}\boldsymbol{I}$. To resolve the kinematic redundancy of the robot for both controllers, we implement a compliant regulator in the nullspace of the end-effector task using the initial configuration as a reference. Both approaches run at a control frequency of 1 kHz.

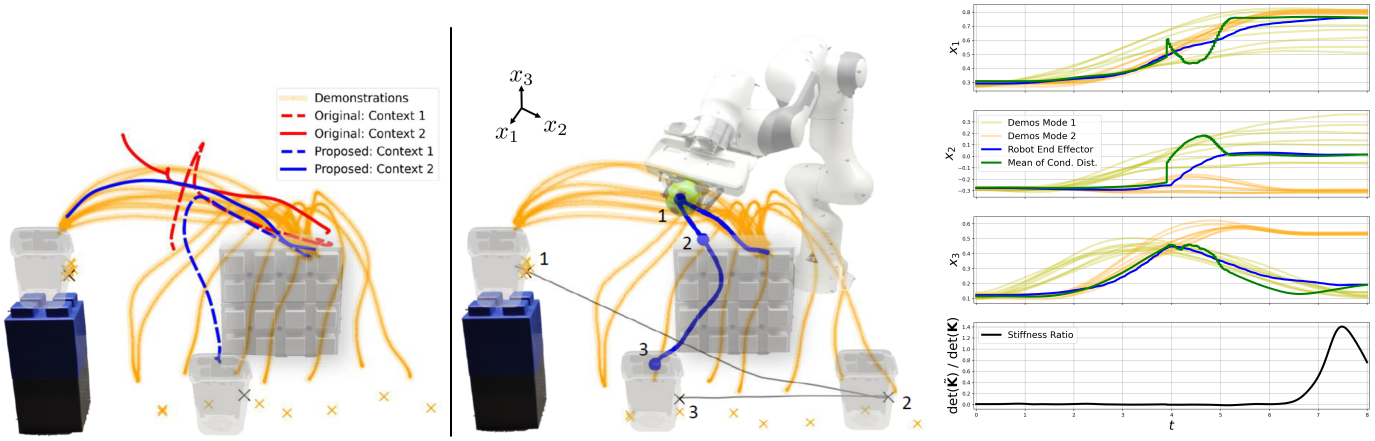

Fig. 5. The Panda robot has to drop a ball into the box using the presented imitation controller (blue). The skill is demonstrated (orange trajectories) for a given box position (orange crosses). Experiment 1 (*left*): The original controller (red) and the proposed controller are tested on two different static box positions (context 1 and 2). Experiment 2 (*center* and *right*): The proposed controller is tested in a dynamic scenario, where the context variable changes during the execution (i.e. the box is moving from position 1 to 3) such that the position reference (green) changes accordingly. The proposed controller adapts its stiffness based on the variability of the demonstrations and based on the correlation of the trajectory phase with the context variable (indicated by the stiffness ratio).

Figure 5 illustrates the results of the experiments (see also the accompanying video). In the left plot, we show a comparison between the proposed controller and the original controller for a box position that does not change during the execution. The dashed lines depict the behavior of the original controller and the proposed controller for the lower box position, while the corresponding solid lines show the behavior for the upper left box position. We can see that both controllers manage to generate motions without the robot colliding with the environment. However, the original controller does not move the robot to the required position, and the robot fails to put the ball into the box because of the inherent perturbations that are not part of the model that both controllers are based on. Our proposed controller, on the other hand, manages to move the robot to successfully drop the ball into the box, for both box positions, as also indicated by the evaluation of the tracking error in Table IV.

TABLE IV
METRIC-BASED COMPARISON FOR THE BALL-IN-BOX TASK (FIG. 5).

Method and Context	**Tracking Error**
Original, Context 1	319.45
Proposed, Context 1	0.46
Original, Context 2	520.76
Proposed, Context 2	0.71

In the second experiment, we evaluate the capability of the proposed controller to adapt robustly to dynamic changes of the context variable, i.e. the box position. The center plot and the right plot in Figure 5 shows the resulting end-effector trajectory produced by the proposed controller. During the execution of the task, the box is moved from position 1 (as labeled in Figure 5) to position 2 then 3, as depicted by the black curve in the center plot. The corresponding markers on the robot trajectory roughly indicate the end-effector positions at that time. The mean of the conditioned trajectory distribution represents the reference of the controller. It can be seen that the movement of the box results in jerky changes of the reference around t=4s, which would be tracked stiffly by a controller with constant feedback gains. The plot in the lower-right corner of Figure 5 illustrates the ratio between the determinant of the varying stiffness gain $\tilde{\boldsymbol{K}}$ and the tuned constant stiffness gain $\boldsymbol{K}$. It shows that the controller learned to use a higher stiffness as the correlation between the context and the trajectory phase increases. In this case, the correlation grows towards the end of the trajectory as the context mainly affects the final position of the end-effector. In summary, the proposed controller solves the given task by switching from the second mode to the first mode after moving the box from the upper left position to a lower position. The resulting trajectory shows that the controller produces smooth transitions between the two modes, together with a smooth adaptation to the changing box position by exploiting the demonstrated variations.

V. CONCLUSION

In this paper, we derived a stochastic feedback controller by imposing a compliant control structure on the latent trajectory feature variable $\boldsymbol{w}$ and conditioning the control action on the demonstrated behavior. We showed that the original controller proposed in [5] is similar to a limit case of our proposed controller. We analyzed the robustness of the closed-loop system depending on the parameters of the proposed controller and compared the results with the original controller. Furthermore, we showed that our proposed controller readily extends to multiple modes and to the adaptation to dynamic context variables in realtime. We evaluated the theoretical hypotheses in simulated and real-world experiments with the Franka Emika Panda robot. We showcased in these experiments that our proposed controller outperforms the original controller in terms of robustness, and that this property is a key to the successful reproduction of demonstrated movements with robots characterized by unmodeled dynamic effects (typically, joint friction).

In future work, we plan to investigate practical limitations of the presented approach, e.g. finding computational bottlenecks

and testing tasks with more than two concurring modes. We also plan to extend the proposed approach to the problem of robustness to context variables that are far from the demonstrated distribution and against intentional physical interactions to consider human-robot interaction in a principled way (e.g. to safely switch to a different mode by interacting with the robot). Furthermore, we plan to investigate adaptation mechanisms for the phase variable of the reference distribution in order to relax the time-dependency of the proposed controller.

APPENDIX A
CARTESIAN SPACE DYNAMICS

The Cartesian space dynamics can be derived by using the instantaneous equalities

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{q}}, \quad \ddot{\boldsymbol{x}} = \boldsymbol{J}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}\dot{\boldsymbol{q}},$$

where $\boldsymbol{x} \in \mathbb{R}^6$ represents the position and orientation of the end-effector in the Cartesian space and $\boldsymbol{J}$ is the corresponding Jacobian matrix. Consequently, the robot dynamics can be reformulated as

$$\boldsymbol{M}_x\ddot{\boldsymbol{x}} + \boldsymbol{C}_x\dot{\boldsymbol{x}} + \boldsymbol{f}_{c,g} = \boldsymbol{f}_x + \boldsymbol{f}_{\text{ext}}.$$

with the Cartesian space dynamic parameters $\boldsymbol{M}_x{=}(\boldsymbol{J}\boldsymbol{M}^{-1}\boldsymbol{J}^\top)^{-1}$, $\boldsymbol{C}_x{=}(\boldsymbol{J}^\dagger)^\top\boldsymbol{C}\boldsymbol{J}^\dagger - \boldsymbol{M}_x\dot{\boldsymbol{J}}\boldsymbol{J}^\dagger$ and $\boldsymbol{f}_{c,g}{=}((\boldsymbol{J}^\dagger)^\top\boldsymbol{C} - \boldsymbol{M}_x\dot{\boldsymbol{J}})(\boldsymbol{I} - \boldsymbol{J}^\dagger\boldsymbol{J})\dot{\boldsymbol{q}} + \boldsymbol{J}^\dagger\boldsymbol{g}$ that all depend on the robot state.

APPENDIX B
CONTROL MEAN FOR INFINITELY SMALL TOLERANCES

In the case of a point mass, the proposed control formulation reduces to

$$\begin{aligned}\boldsymbol{\mu}_{\boldsymbol{u}} &= -\boldsymbol{K}(\boldsymbol{q} - \boldsymbol{\Phi}\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}}) - \boldsymbol{D}(\dot{\boldsymbol{q}} - \dot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}}) + \boldsymbol{M}\ddot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}} \\ &= -\boldsymbol{K}_{\boldsymbol{y}}(\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}}) + \boldsymbol{M}\ddot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}},\end{aligned}$$

where $\boldsymbol{K}_{\boldsymbol{y}}$ is a block diagonal matrix composed of $\boldsymbol{K}$ and $\boldsymbol{D}$. Following the notion using the compact robot state $\boldsymbol{y}{=}[\boldsymbol{q}^\top, \dot{\boldsymbol{q}}^\top]^\top$, the mean vector of the conditional weight distribution can be written as

$$\begin{aligned}\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}} &= \boldsymbol{\mu}_{\boldsymbol{w}} + (\boldsymbol{\Sigma}_{\boldsymbol{w}}^{-1} + \boldsymbol{\Psi}^\top\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}\boldsymbol{\Psi})^{-1}\boldsymbol{\Psi}^\top\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{\mu}_{\boldsymbol{w}}) \\ &= \boldsymbol{\mu}_{\boldsymbol{w}} + (\boldsymbol{I} + \boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}\boldsymbol{\Psi})^{-1}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{\mu}_{\boldsymbol{w}}),\end{aligned}$$

where $\boldsymbol{\Sigma}_{\boldsymbol{y}}$ is a block diagonal matrix composed of $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ and $\boldsymbol{\Sigma}_{\dot{\boldsymbol{q}}}$. Next, we use the equality $(\boldsymbol{I} + \boldsymbol{A}\boldsymbol{B})^{-1}\boldsymbol{A}{=}\boldsymbol{A}(\boldsymbol{I} + \boldsymbol{B}\boldsymbol{A})^{-1}$, that can be found in [20], to rewrite the previous result as

$$\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}} = \boldsymbol{\mu}_{\boldsymbol{w}} + \boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}(\boldsymbol{I} + \boldsymbol{\Psi}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1})^{-1}(\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{\mu}_{\boldsymbol{w}}).$$

In the limit case of infinitely small eigenvalues of a scaled identity tolerance matrix $\boldsymbol{\Sigma}_{\boldsymbol{y}}{=}\sigma_{\boldsymbol{y}}\boldsymbol{I}$, the mean of the conditional distribution can be written as

$$\begin{aligned}&\lim_{\sigma_{\boldsymbol{y}}\to 0}\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}} = \\ &\lim_{\sigma_{\boldsymbol{y}}\to 0}\boldsymbol{\mu}_{\boldsymbol{w}} + \boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top(\sigma_{\boldsymbol{y}}\boldsymbol{I} + \boldsymbol{\Psi}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top)^{-1}(\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{\mu}_{\boldsymbol{w}}) = \\ &\qquad \boldsymbol{\mu}_{\boldsymbol{w}} + \boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top(\boldsymbol{\Psi}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top)^{-1}(\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{\mu}_{\boldsymbol{w}}).\end{aligned}$$

By inserting this result into the controller and knowing that $\lim_{\sigma_{\boldsymbol{y}}\to 0}\boldsymbol{\Psi}\boldsymbol{\mu}_{\boldsymbol{w}|\boldsymbol{y}}{=}\boldsymbol{y}$, we obtain the final result of the derivation

$$\lim_{\sigma_{\boldsymbol{y}}\to 0}\boldsymbol{\mu}_{\boldsymbol{u}} = \boldsymbol{M}\ddot{\boldsymbol{\Phi}}\boldsymbol{\mu}_{\boldsymbol{w}} + \boldsymbol{M}\ddot{\boldsymbol{\Phi}}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top(\boldsymbol{\Psi}\boldsymbol{\Sigma}_{\boldsymbol{w}}\boldsymbol{\Psi}^\top)^{-1}(\boldsymbol{y} - \boldsymbol{\Psi}\boldsymbol{\mu}_{\boldsymbol{w}}).$$

REFERENCES

[1] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *Foundations and Trends® in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

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

[3] ——, "Learning control Lyapunov function to ensure stability of dynamical system-based robot reaching motions," *Robotics and Autonomous Systems*, vol. 62, no. 6, pp. 752–765, 2014.

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

[5] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, "Using probabilistic movement primitives in robotics," *Autonomous Robots*, vol. 42, no. 3, pp. 529–551, 2018.

[6] A. Paraschos, E. Rueckert, J. Peters, and G. Neumann, "Probabilistic movement primitives under unknown system dynamics," *Advanced Robotics*, vol. 32, no. 6, pp. 297–310, 2018.

[7] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Learning attractor landscapes for learning motor primitives," in *Advances in Neural Information Processing Systems (NIPS)*, 2002, pp. 1547–1554.

[8] M. Ewerton, G. Neumann, R. Lioutikov, H. Ben Amor, J. Peters, and G. Maeda, "Learning multiple collaborative tasks with a mixture of interaction primitives," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2015, pp. 1535–1542.

[9] 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.

[10] Y. Zhou, J. Gao, and T. Asfour, "Movement primitive learning and generalization: Using mixture density networks," *IEEE Robotics Automation Magazine*, vol. 27, no. 2, pp. 22–32, 2020.

[11] M. Khoramshahi and A. Billard, "A dynamical system approach for detection and reaction to human guidance in physical human-robot interaction," *Autonomous Robots*, vol. 44, no. 8, pp. 1411–1429, 2020.

[12] M. Ewerton, O. Arenz, G. Maeda, D. Koert, Z. Kolev, M. Takahashi, and J. Peters, "Learning trajectory distributions for assisted teleoperation and path planning," *Frontiers in Robotics and AI*, vol. 6, p. 89, 2019.

[13] F. Brandherm, J. Peters, G. Neumann, and R. Akrour, "Learning replanning policies with direct policy search," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2196–2203, 2019.

[14] S. Calinon and D. Lee, "Learning control," in *Humanoid Robotics: a Reference*, P. Vadakkepat and A. Goswami, Eds. Springer, 2019, pp. 1261–1312.

[15] S. Calinon, "Mixture models for the analysis, edition, and synthesis of continuous time series," in *Mixture Models and Applications*, N. Bouguila and W. Fan, Eds. Springer, 2019, pp. 39–57.

[16] T. Kulak, J. Silvério, and S. Calinon, "Fourier movement primitives: an approach for learning rhythmic robot skills from demonstrations," in *Proc. Robotics: Science and Systems (RSS)*, 2020.

[17] S. Gomez-Gonzalez, G. Neumann, B. Schölkopf, and J. Peters, "Adaptation and robust learning of probabilistic movement primitives," *IEEE Transactions on Robotics (T-Ro)*, vol. 36, no. 2, pp. 366–379, 2020.

[18] E. Pignat and S. Calinon, "Bayesian Gaussian mixture model for robotic policy imitation," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4452–4458, 2019.

[19] E. Pignat, J. Silvério, and S. Calinon, "Learning from demonstration using products of experts: Applications to manipulation and task prioritization," *arXiv:2010.03505*, 2020.

[20] S. R. Searle, *Matrix Algebra Useful for Statistics*. John Wiley and Sons, 1982.