Assisted Teleoperation in Changing Environments with a Mixture of Virtual Guides

Marco Ewerton[a, b], Oleg Arenz[a], and Jan Peters[a, c]

[a]Intelligent Autonomous Systems Group, Department of Computer Science, Technische Universität Darmstadt, Hochschulstr. 10, 64289 Darmstadt, Germany; [b] Perception and Activity Understanding Group / Robot Learning and Interaction Group, Idiap Research Institute, Rue Marconi 19, 1920 Martigny, Switzerland; [c]Max Planck Institute for Intelligent Systems, Max-Planck-Ring 4, 72076 Tübingen, Germany

ABSTRACT
Haptic guidance is a powerful technique to combine the strengths of humans and autonomous systems for teleoperation. The autonomous system can provide haptic cues to enable the operator to perform precise movements; the operator can interfere with the plan of the autonomous system leveraging his/her superior cognitive capabilities. However, providing haptic cues such that the individual strengths are not impaired is challenging because low forces provide little guidance, whereas strong forces can hinder the operator in realizing his/her plan. Based on variational inference, we learn a Gaussian mixture model (GMM) over trajectories to accomplish a given task. The learned GMM is used to construct a potential field which determines the haptic cues. The potential field smoothly changes during teleoperation based on our updated belief over the plans and their respective phases. Furthermore, new plans are learned online when the operator does not follow any of the proposed plans, or after changes in the environment. User studies confirm that our framework helps users perform teleoperation tasks more accurately than without haptic cues and, in some cases, faster. Moreover, we demonstrate the use of our framework to help a subject teleoperate a 7 DoF manipulator in a pick-and-place task.

1. Introduction

Robots are powerful tools for solving tasks that would be very laborious for humans. Furthermore, they can be deployed in hazardous environments, e.g., for sorting nuclear waste [1]. However, especially when operating in unstructured environments, fully autonomous robots may not provide the required reliability. On the other hand, manually controlling the robots, for example via teleoperation, is often cumbersome and inefficient. Hence, there is great interest in shared autonomy, where both, the human operator and the autonomous system, influence the robot's actions [2].

In safety-critical environments, it is often desirable that the operator remains in full control of the robot. For example, by teleoperation of a robot with a master haptic

device [3], the autonomous system should apply a wrench on the handle that is large enough to provide guidance, yet weaker than the operator's wrench. However, the haptic guidance can be inconvenient, especially if the plans of both agents are badly aligned, or if the haptic guidance changes suddenly.

A way of avoiding a misalignment between the plans of the human and of the autonomous system is to vary the stiffness of the master device according to the required precision at each state. This way, the user can easily deviate from the plan proposed by the autonomous system when he/she sees fit as long as this deviation does not compromise the necessary accuracy to solve the task. For example, the robot may need to be controlled with high precision while picking up a small item, whereas the precision

Figure 1. User operating the haptic device Haption Virtuose 6D TAO to perform teleoperation tasks.

while moving towards that item can be rather low since the intention of the human might be still unclear. An autonomous system that takes into account these variabilities while solving a task could apply relatively high wrenches while picking up the small item and restrict itself to merely providing cues while guiding the operator towards that object, enabling the operator to move towards another object for example. In order to encode these variabilities, probabilistic trajectory representations are often used [2,4].

Gaussian distributions over trajectories, such as probabilistic movement primitives (ProMPs) [5], are particularly convenient because their mean can correspond to a reference trajectory and their covariance matrix can encode the variability along that trajectory. However, a single ProMP only encodes a single (noisy) plan and may thus fail to provide adequate haptic feedback if several plans are feasible. Instead, the autonomous system should consider several possible plans and the possibility of none of its plans being in accordance with the operator's intentions.

We address these challenges by providing haptic feedback based on a mixture of ProMPs, where one of the ProMPs is fixed and has very high variance for all time steps. This ProMP corresponds to a "freelance" plan which induces negligible wrenches. The remaining ProMPs are the mixture components

Figure 2. Diagram of our proposed framework for assisted teleoperation. First, we define a reward function according to the task at hand (examples in Section 7). Then, variational inference is used to learn a mixture of virtual guides (Section 3). Bayes filtering is used to infer the user intention, i.e., the plan and the phase (stage of the movement) intended by the user (Section 5). Finally, this inference and the gradient of the potential energy based on the learned mixture modulate the force feedback provided by the haptic device to the user (Section 4).

of a Gaussian mixture model (GMM) over trajectories which is learned in an episodic maximum entropy reinforcement learning setting. In this work, a method for variational inference, VIPS [6], is used to learn this GMM.
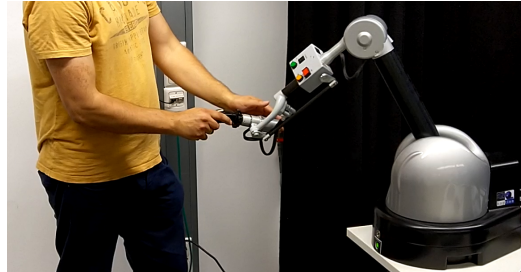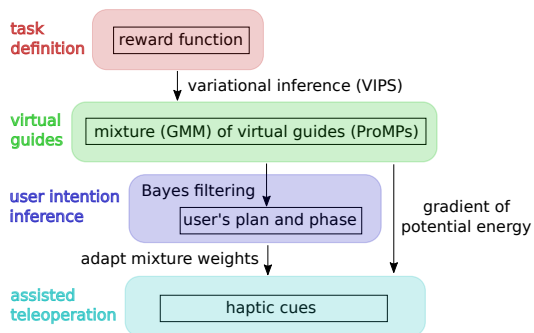
We update our belief over the plan that the operator is following, as well as its phase (the stage of execution of the movement), based on the operator's actions. If the operator does not seem to follow any of the plans of the autonomous system, our framework assigns a high probability to the freelance plan and the operator will receive low haptic feedback. In that case, we can plan a new trajectory online and smoothly blend it in. Furthermore, plans that get invalidated due to changes in the environment can be detected and removed.

In order to provide haptic feedback based on our current belief of the operator's intention, we construct a potential field where each discretized phase of each plan is represented by the energy of a Gaussian distribution in end-effector space and use its gradient to provide haptic feedback. The potential field is thus given by the negative log probability density of a GMM, where the component's weights are given by our belief, and where each plan creates a valley along which the operator is guided. Fig. 2 provides an overview of the modules of our proposed framework and how they are combined to achieve assisted teleoperation.

The efficacy of the proposed method is validated by experiments in which users perform teleoperation tasks by manipulating a haptic device (see Fig. 1). One of the tasks consists of teleoperating a 7 DoF robot arm to pick up objects. The human can switch between different assistive guides and our system helps the user pick up objects accurately. The second task consists of translating and rotating a pole in a virtual environment to reach a certain pose while avoiding obstacles. Both tasks have multiple possible solutions, which are learned by our method.

2. Connection to Prior Work

2.1. Potential Fields and Dynamical Systems

Potential fields have long found applications in shared control tasks. Aigner and Mc-Carragher [7] use potential fields to drive a robot arm away from obstacles and from the borders of the workspace as well as to let the human change the autonomous behavior of the robot.

Potential fields can be used to ease the teleoperation of a robot to grasp small objects. Howard and Park [8] describe an experiment where a camera has been mounted on the gripper of a robot. Potential fields help the user move the gripper such that the object appears at the center of the image captured by the camera, meaning that the gripper is just above the object.

Another application of potential fields designed to help users in teleoperation tasks is shown by Gioioso et al. [9]. In that work, a control loop is proposed to assist the teleoperation of a quadrotor both in contact-free flight and when applying forces to objects.

In contrast to the mentioned works, our approach consists of learning potential fields that provide the user with one or more paths to solve a task. Furthermore, our approach can deal with obstacles without getting stuck in local minima as it is often the case in classical potential fields methods.

Dynamical systems approaches more sophisticated than classical potential fields have recently been explored in tasks involving avoidance of obstacles of various shapes and interaction with humans. Huber et al. [10] have presented an approach to make the end-effector of a robot avoid convex and star-shaped obstacles while moving towards an attractor. That approach consists of multiplying the original linear dynamical

system by a dynamic modulation matrix, which can be computed in closed form to guarantee the impenetrability of convex and star-shaped obstacles. Our work does not provide the same strong guarantees. On the other hand, it finds multimodal trajectory distributions to solve planning problems with arbitrary reward functions, which can be used as guiding virtual fixtures in assisted teleoperation tasks.

Potential fields can be learned from demonstrations, which increases the applicability of robots beyond hard-coded behaviors, and can lead to safe and reliable movements in unstructured environments. Khansari-Zadeh and Khatib [11] propose a method to learn a potential energy function and a dissipative field from demonstrations. The control policy is the negative gradient of the potential function minus the dissipative field. Movements with a single target are addressed. Our work bears similarities to [11] in the sense that our approach also involves learning a potential field. Nevertheless, in our work, the objective is not necessarily to drive the robot along a certain path to the target. Instead, we address the problem of giving force feedback to a user in assisted teleoperation tasks. There may be several solutions to the task and the user can decide to switch from a path to another. Moreover, our potential field is learned through reinforcement learning instead of from demonstrations. This feature is especially desirable when it is hard for the human to give suitable demonstrations.

2.2. Gaussian Mixture Models

Ewerton et al. [12] propose learning a Gaussian mixture model (GMM) of trajectories from demonstrations to model human-robot collaboration tasks. This approach is extended by Koert et al. [13] to enable the open-ended learning of a skill library for collaborative tasks. The open-ended learning is achieved through the use of incremental GMMs [14], which do not require pre-specifying the number of mixture components. In this paper, a variational inference method (VIPS) to learn GMMs has been employed [6]. VIPS requires specifying a maximum number of components instead of the exact number and is used to learn GMMs given a reward function instead of demonstrations.

Raiola et al. [15] also learn GMMs of trajectories from demonstrations. In their work, virtual mechanisms attract the end-effector of the robot like spring-damper systems, guiding the user in co-manipulation tasks. The GMM formulation allows for the existence of multiple virtual mechanisms. In our work, the GMMs of trajectories are learned through reinforcement learning, which can be very helpful when the user cannot provide suitable demonstrations, e.g. in a challenging teleoperation task with several degrees of freedom.

2.3. Variational Inference

Pignat et al. [16] propose a method to learn a mixture of models through variational inference. In that work, a mixture of Gaussians is learned to cover a certain 2D space while avoiding obstacles. Subsequently, a shortest path algorithm is used to find a sequence of mixture components from a given start position to a given end position. Covering the entire workspace with mixture components does not scale very well for higher dimensions. Therefore, in our work, the components of the learned GMM correspond directly to successful distributions of trajectories to solve the task instead of representing clusters of successful poses.

3. Learning a Mixture of ProMPs

Teleoperation tasks usually accept multiple solutions. As exemplified in Section 7, it may be possible to move around obstacles using different plans and there may be multiple targets. Besides, some parts of a successful plan may allow for larger variability than others. In order to encode these variabilities, we want to represent each plan by a trajectory distribution with phase-dependent variance as illustrated in Fig. 3. We, therefore, represent each plan as a probabilistic movement primitive (ProMP) [5]. A ProMP is a Gaussian distribution of trajectories. It can be seen as a Gaussian distribution both in the space of poses as well as in the space of trajectory parameters. This representation is convenient for our needs since the variance of this Gaussian can be used to modulate the stiffness of an autonomous system in a shared autonomy setting, allowing for more or less variability at each stage of

Figure 3. Mixture of three trajectory distributions (ProMPs) learned in a 2D scenario. The blue lines represent the sides of walls. A point particle needs to move from a start position (green ×) to an end position (red ×) while avoiding the walls. VIPS can learn a mixture of ProMPs to solve this task. The ellipses represent Gaussians in Cartesian space, which are used to construct a potential field. This potential field is used to give the user haptic cues as explained in detail in Section 4. The number of ellipses, i.e., the number of Gaussians in Cartesian space depends on our phase discretization.

a plan. Moreover, ProMPs provide a compact representation of trajectories, which is helpful for reinforcement learning algorithms.

ProMPs represent a trajectory of end-effector poses $\mathbf{x}(\nu) \in \mathbb{R}^n$, where n is the number of degrees of freedom of the end-effector, as a function $\mathbf{x_w}(\nu) = \mathbf{\Phi}(\nu)\mathbf{w}$ that is linear in normalized Gaussian radial basis functions evenly spaced along the movement phase $\nu \in [0,1]$. The matrix $\mathbf{\Phi}(\nu) \in \mathbb{R}^{n \times mn}$, where m is the number of basis functions, is a block-diagonal matrix

$$\mathbf{\Phi}(\nu) = \begin{bmatrix} \boldsymbol{\phi}(\nu)^\top & & \\ & \ddots & \\ & & \boldsymbol{\phi}(\nu)^\top \end{bmatrix}. \tag{1}$$

Each element of the vector $\boldsymbol{\phi}(\nu) \in \mathbb{R}^m$ is a normalized Gaussian radial basis function of the form

$$\phi_c(\nu) = \frac{\exp\left(-\frac{1}{2}\left(\nu - \frac{c-1}{m-1}\right)^2\right)}{\sum_{c=1}^{m} \exp\left(-\frac{1}{2}\left(\nu - \frac{c-1}{m-1}\right)^2\right)} \tag{2}$$

evaluated at ν, where the variable $c \in \{1, 2, \cdots, m\}$ specifies the center of each basis function.

The trajectory distributions (ProMPs) depicted in Fig. 3, for instance, have $m = 10$ radial basis functions and $n = 2$ in this case. Since the pose $\mathbf{x_w}(\nu)$ is an affine
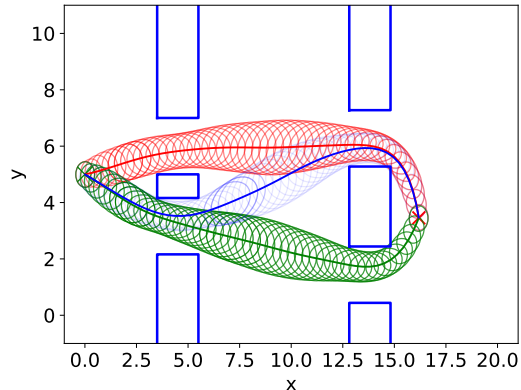
function of the weight vector $\mathbf{w} \in \mathbb{R}^{mn}$, a Gaussian distribution over weights $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w)$ induces a Gaussian distribution over poses

$$p(\mathbf{x}|\nu) = \mathcal{N}\left(\mathbf{x}|\boldsymbol{\Phi}(\nu)\boldsymbol{\mu}_w, \boldsymbol{\Phi}(\nu)\boldsymbol{\Sigma}_w\boldsymbol{\Phi}(\nu)^\top\right). \tag{3}$$

Hence, a mixture of ProMPs can be represented by a GMM over weights

$$p(\mathbf{w}) = \sum_{o=1}^{N_o} p(o)p(\mathbf{w}|o), \tag{4}$$

where a multinomial distribution $p(o)$ assigns a normalized weight to each option $p(\mathbf{w}|o) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_o, \boldsymbol{\Sigma}_o)$ and N_o is the number of options. Each option represents a different plan. Figs. 3 and 6, for example, show situations where three options have been learned.

We consider a reinforcement learning setting with a reward function $r(\mathbf{x_w}(\nu), \nu)$ that depends on the end-effector pose $\mathbf{x}$ and the phase ν. Such a reward function can, for example, assume higher values when close to desired poses or when the acceleration is low and lower values when close to obstacles or when the acceleration is high. We aim to learn a mixture of ProMPs that achieves high episodic reward $r(\mathbf{w}) = \int_0^1 r(\mathbf{x_w}(\nu), \nu)d\nu$ in expectation while maintaining high entropy. These assumptions can be phrased as the optimization problem

$$\underset{p(o),\boldsymbol{\mu}_o,\boldsymbol{\Sigma}_o}{\arg\max} \sum_{o=1}^{N_o} p(o) \int p(\mathbf{w}|o)r(\mathbf{w})d\mathbf{w} + H(p(\mathbf{w})) \tag{5}$$

with the Shannon entropy $H(p(\mathbf{w}))$ of the GMM. The entropy objective rewards variability and is crucial to prevent convergence to a single trajectory. Please note that scaling the reward function affects the optimal solution due to the entropy objective. We assume that the reward function is adequately scaled to result in sufficient variability without inducing undesirable trajectories.

As we consider an episodic setting, that is, we do not assume nor exploit access to time-series data, the maximum entropy reinforcement learning problem (5) can be equivalently framed as variational inference. The connection between this reinforcement learning formulation and variational inference can be shown by introducing a reward distribution

$$p_r(\mathbf{w}) = \frac{1}{Z_r} \exp(r(\mathbf{w})) \tag{6}$$

with partition function $Z_r = \int \exp(r(\mathbf{w}))d\mathbf{w}$. The optimization problem (5) can now be reformulated as the variational inference problem of approximating the reward distribution $p_r(\mathbf{w})$ with a GMM $p(\mathbf{w})$ by minimizing the reverse Kullback-Leibler

divergence $D_{\mathrm{KL}}(p(\mathbf{w})||p_r(\mathbf{w}))$. This optimization problem is

$$
\arg\min_{p(o),\boldsymbol{\mu}_o,\boldsymbol{\Sigma}_o} D_{\mathrm{KL}}(p(\mathbf{w})||p_r(\mathbf{w}))
$$

$$
= \arg\max_{p(o),\boldsymbol{\mu}_o,\boldsymbol{\Sigma}_o} \int p(\mathbf{w}) \log p_r(\mathbf{w}) d\mathbf{w} + H(p(\mathbf{w}))
$$

$$
= \arg\max_{p(o),\boldsymbol{\mu}_o,\boldsymbol{\Sigma}_o} \int p(\mathbf{w}) r(\mathbf{w}) d\mathbf{w} - \log Z_r + H(p(\mathbf{w})). \tag{7}
$$

As the constant partition function Z_r does not affect the solution of the optimization problem, (5) and (7) are equivalent. Therefore, we can learn a mixture of ProMPs with any variational inference method that is capable of learning GMMs. We use variational inference by policy search (VIPS) [6]. Fig. 3 shows a mixture of three ProMPs learned by VIPS to solve a planning problem in a 2D scenario.

4. Computing the Haptic Cues

Haptic cues to the operator are provided by a potential field in end-effector space based on our GMM in weight space. More precisely, we discretize the phase ν in T_o steps for each plan, denoted by $\nu_1, \ldots \nu_{T_o}$, and compute the marginal distribution

$$
p(\mathbf{x}) = \sum_{o=1}^{N_o} p(o) \sum_{i=1}^{T_o} p(\nu_i|o) p(\mathbf{x}|\nu_i, o)
$$

$$
= \sum_{o=1}^{N_o} p(o) \sum_{i=1}^{T_o} p(\nu_i|o) \mathcal{N}(\mathbf{x}|\boldsymbol{\Phi}(\boldsymbol{\nu}_i)\boldsymbol{\mu}_o, \boldsymbol{\Phi}(\boldsymbol{\nu}_i)\boldsymbol{\Sigma}_o \boldsymbol{\Phi}(\nu_i)^\top)
$$

$$
= \sum_{o=1}^{N_o} p(o) \sum_{i=1}^{T_o} p(\nu_i|o) \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{o,i}, \boldsymbol{\Sigma}_{o,i}) \tag{8}
$$

using our belief $p(\nu_i|o)$ about the current phase of each plan o and the distribution $p(\mathbf{x}|\nu_i, o)$ over end-effector poses at phase ν_i in plan o. Each ellipse in Fig. 3 corresponds to a distribution $p(\mathbf{x}|\nu_i, o)$. We will discuss in Section 5 how we adapt online the belief $p(\nu_i|o)$ over the phase and also the belief $p(o)$ over the plan.

The marginal distribution $p(\mathbf{x})$ corresponds to a GMM with components for each plan and each phase. Similarly to Hamiltonian MCMC [17], we define a potential energy $E(\mathbf{x}) = -\log p(\mathbf{x})$ based on the negative log of the marginal, which induces a wrench that is given by the negative gradient of the energy

$$
\boldsymbol{\tau}(\mathbf{x}) = -\frac{\partial E(\mathbf{x})}{\partial \mathbf{x}} = \frac{\partial \log p(\mathbf{x})}{\partial \mathbf{x}}. \tag{9}
$$

By applying the wrench $\boldsymbol{\tau}(\mathbf{x})$, we guide the operator towards low energy regions based on the marginal $p(\mathbf{x})$. As the marginal is computed based on an approximation of the episodic reward distribution $p_r(\mathbf{w})$, the operator is guided towards trajectories that provide a high episodic reward. Please note that directly using the negative reward function $-r(\mathbf{x}, \nu)$ for constructing a potential field would guide the operator towards regions of high immediate reward, whereas our learned, planning-based potential field

assists the operator in achieving a high episodic reward. For example, consider a goal position that is directly behind a wall and a reward function that penalizes the distance to the goal and close proximity to the wall. Whereas the immediate reward results in a potential field that would keep the operator in the current position, our learned potential field would create several valleys along which the operator is guided around the wall. Furthermore, the energy of the marginal distribution $p(\mathbf{x})$ corresponds to the negative log of a GMM and is thus differentiable and smooth. A smooth potential field is highly desirable in order to avoid sudden jumps in the direction or magnitude of the wrench that is applied to the handle of the master device. The gradient of the log of the marginal is

$$\frac{\partial \log p(\mathbf{x})}{\partial \mathbf{x}} = \sum_{o=1}^{N_o} \sum_{i=1}^{T_o} p(o, \nu_i | \mathbf{x}) \Sigma_{o,i}^{-1} (\mu_{o,i} - \mathbf{x}). \tag{10}$$

Hence, each component attracts the operator with a force of magnitude proportional to the distance to its mean and weighted by its responsibility

$$p(o, \nu_i | \mathbf{x}) = \frac{p(o) p(\nu_i | o) p(\mathbf{x} | \nu_i, o)}{p(\mathbf{x})}. \tag{11}$$

The operator would typically get small haptic cues if he/she is close to one of the components because, due to the proximity to that component, its responsibility would often be close to one and its force contribution would be close to zero. However, if the operator is far from every component, all components would have large force contributions and their weighted average would typically also be large. Such behavior can be desirable if we do not want to allow the operator to leave the planned trajectories.

However, if the operator should keep full control, such behavior would be undesirable even if the maximum wrench gets capped. In such cases, we can add an additional component with large variance to our mixture model that covers the whole workspace. Such a component corresponds to a freelance plan with a single phase, where the operator can move freely within the workspace. Due to its high variance, this component always has low force contributions and gets low responsibility if the operator is close to the planning based components, but high responsibility if the operator is far from any planned components.

Without the user, when assuming the beliefs $p(o)$ and $p(\nu_i | o)$ to be fixed and the control rate at which the haptic cues are updated to be infinite, no energy gets injected into our system. We include a damping term, such that the total wrench is

$$\boldsymbol{\tau}_{\text{total}}(\mathbf{x}, \dot{\mathbf{x}}) = \boldsymbol{\tau}(\mathbf{x}) - k_{\text{damp}} \dot{\mathbf{x}}. \tag{12}$$

By including such a damping, the total energy of the system always decreases, which leads to its stability. The damping term also prevents oscillatory force feedback close to low-variance components that were caused by our limited control rate.

5. Adapting the Mixture Weights by Updating the Belief

In order to assist the user in a teleoperation task, our system needs to infer from the pose of the handle of the haptic device commanded by the human which ProMP (also
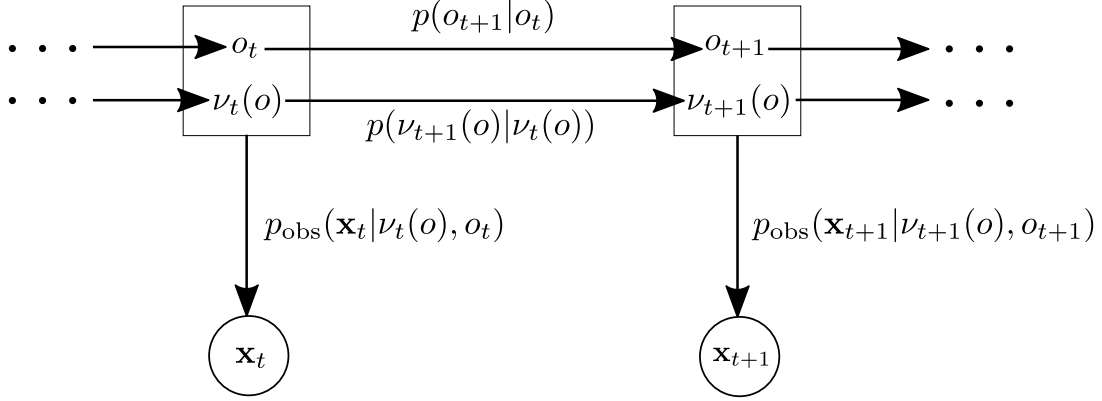
Figure 4. Hidden Markov model (HMM) underlying the update of the belief about the plan o and phase ν.

referred to as option or plan) the human intends to follow. We may also want to guide the human forward along the intended plan while enabling him/her to stop or move backward. Therefore, we may also need to infer the intended phase along the plan of interest.

These inference problems are formalized by using the weights $p(o)$ and $p(\nu|o)$ to reflect our belief about the plan o that the operator is currently employing and about the phase ν along each plan. We initialize $p(o)$ based on the weights that were learned by VIPS, which assigns higher weights to components that achieve a higher expected reward, higher entropy or that differ from the remaining components. We initialize our belief $p(\nu|o)$ over the phase as a uniform distribution.

In order to update our belief during teleoperation, we formulate a hidden Markov model (HMM) where the true plan o_t of the operator and the phases $\nu_t(o)$ at time t are hidden variables. Please note that t corresponds to the discretized real time that increments at the control rate at which we send the desired wrench to the haptic device. Fig. 4 represents this HMM. The transition probabilities $p(\nu_{t+1}(o)|\nu_t(o))$ and $p(o_{t+1}|o_t)$ are explained in Sections 5.1 and 5.2, respectively. The emission probability $p_{\text{obs}}(\mathbf{x}_t|\nu_t(o), o_t)$ is obtained by rescaling the Covariance matrix of $p(\mathbf{x}_t|\nu_t(o), o_t)$ to allow for a larger uncertainty in the observation $\mathbf{x}_t$. The rescaling factor for the Covariance matrix is chosen such that the belief about the plan $p(o)$ does not converge too fast in the face of new observations.

5.1. Updating the Belief about the Phase

For modeling the transition probability of the phase $p(\nu_{t+1}(o)|\nu_t(o))$, we assume that the phase progresses by $\Delta\nu$ discrete steps with probability p_{progress} and changes to a random phase otherwise. A prior belief about the phase at time step t can thus be computed based on the posterior belief at time step $t-1$,

$$p_{\text{prior,t}}(\nu|o) = p_{\text{progress}} \mathcal{T}_{\Delta\nu}\big(p_{\text{post,t-1}}(\nu|o)\big) + (1 - p_{\text{progress}})\frac{1}{T_o}, \tag{13}$$

where the shifting operator $\mathcal{T}_{\Delta\nu}(p)$ shifts the probability mass of the distribution p by $\Delta\nu$ forwards in time whereby no mass gets shifted beyond the last phase ν_{T_o}. Please note that $\Delta\nu$ can be any positive real value by shifting fractions of the probability mass of the distribution p. For example, when shifting by $\Delta\nu = 1.5$, half of the probability
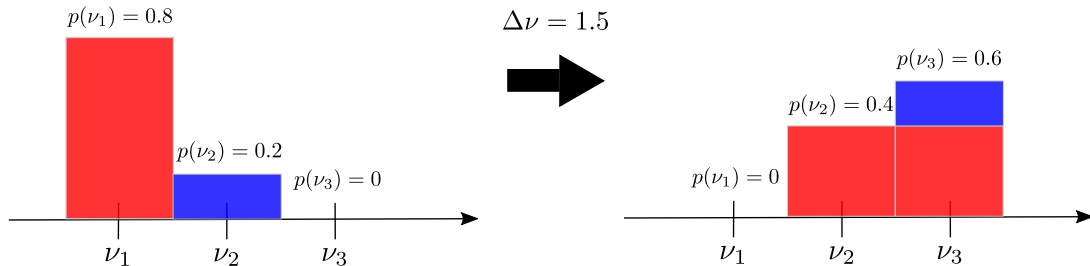
9

Figure 5. Example of change in the probability distribution $p(\nu)$ induced by the shifting operator $\mathcal{T}_{\Delta\nu}(p)$ when the phase is discretized in only three possible values for simplification. With $\Delta\nu = 1.5$, half of the probability mass of ν_1 gets assigned to ν_2 and the other half gets assigned to ν_3. Because ν_3 is the last phase, all the probability mass of ν_2 gets shifted to ν_3.

at ν_1 would get assigned to ν_2 and the other half would get assigned to ν_3. When shifting by $\Delta\nu = 2.0$, the whole probability at ν_1 would get assigned to ν_3. Fig. 5 presents a visualization of the effects of the shifting operator $\mathcal{T}_{\Delta\nu}(p)$ for a case where $T_o = 3$. A formal definition of $\mathcal{T}_{\Delta\nu}(p)$ is provided in Appendix A.

Please note that the progress of the continuous phase, $\delta_\nu(T_o) = \frac{\Delta\nu}{T_o}$, also depends on the number of discretizations for a given plan. Hence, the progress for a given plan can be individually controlled by changing the number of discretizations, T_o. For all our experiments, we chose the same number of discretizations for all plans (except for the stationary freelance-plan) because we do not want different plans to progress with different speed. However, it would be a natural extension to adapt the number of discretizations, for example, depending on the length of the planned (mean) trajectory.

Based on recursive Bayesian estimation, the posterior belief at time t is

$$p_{\text{post,t}}(\nu|o) \propto p_{\text{prior,t}}(\nu|o)p_{\text{obs}}(\mathbf{x}_t|\nu, o). \tag{14}$$

However, when providing haptic cues based on the posterior belief of the phase at time step t, we would not provide any incentive to the operator to progress with the plan. Instead, we compute the haptic cues based on the prior belief of the phase at the next time step, $p_{\text{prior,t+1}}(\nu|o)$. As we assume the phase to progress by $\Delta\nu$ discrete steps at the control rate, computing the negative energy gradient (10) based on the prior belief of the next time step results into a wrench that tends to pull the operator towards the upcoming components for each plan, providing an incentive to progress.

5.2. Updating the Belief about the Plan

We model the transition probability of the plan as

$$p(o|o') = \begin{cases} (1 - p_{\text{switch}}), & \text{if } o' = o \\ p_{\text{switch}}/(N_o - 1), & \text{otherwise} \end{cases} \tag{15}$$

with a small probability p_{switch} of changing the plan and N_o corresponding to the number of plans. We therefore assume that the operator likely follows the same plan as during the last time step, but with low probability might change to another random

plan. The Bayesian belief update

$$p_{\text{post,t}}(o) \propto p_{\text{prior,t}}(o)p(\mathbf{x}_t|o)$$

$$\propto \int p_{t-1}(o')p(o|o')do' \sum_{i=1}^{T_o} p_t(\nu_i|o)p_{\text{obs}}(\mathbf{x}_t|\nu_i, o) \qquad (16)$$

corresponds to the posterior belief of the plan that is pursued by the operator at time t.

6. Adapting the Plans Online

The computational time required to plan a mixture of ProMPs with the proposed method depends strongly on the number of degrees of freedom and basis functions, as well as on the number of components $p(\mathbf{x}|o)$ that are to be learned. VIPS [6] can learn components with full covariance matrices Σ_o; however, to speed up the online replanning of ProMPs, we only learn diagonal covariance matrices. This simplification ignores correlations between different phases and between different degrees of freedoms, resulting in mixture components $p(\mathbf{x}_t|\nu, o)$ in the end-effector space that are axis-aligned.

We found VIPS to be sufficiently efficient for learning small mixtures of ProMPs online. We consider two kinds of events that trigger online replanning, namely, replanning due to changes in the environment and replanning if the operator does not seem to pursue any of the existing plans.

Changes of the environment can be detected for example via vision and typically invalidate the previous plans. We, therefore, remove the previous plans and provide haptic cues only based on the freelance plan and the new plans.

We can detect that the operator is not pursuing any of the current plans based on the belief $p(o_{\text{freelance}})$ of the freelance plan. If the operator does not seem to pursue any of the previous plans, we do not need to delete the previous plans. In order to avoid sudden changes in the haptic cues, we add the new plans with negligible initial weight to the mixture of ProMPs. The weight of the new plans will typically smoothly increase due to the belief updates.

7. Experiments

Two experiments demonstrate applications of our framework to assist humans to perform teleoperation tasks. The haptic device used in these experiments, the Virtuose 6D TAO manufactured by the company Haption [18], is depicted in Fig. 1. The first experiment shows that our framework can help users teleoperate a 7 DoF robot arm. In the second experiment, users need to control the 6D pose of a pole in a virtual environment. User studies validate the efficacy of our framework. The user studies have been conducted in the second experiment to avoid the risk of damaging real equipment as it could be the case due to collisions with objects in the environment of the teleoperated robot arm. Future user studies shall be conducted in tasks involving the teleoperation of real robot arms.

The ProMPs used in these experiments have $m = 7$ basis functions. We have chosen a number of basis functions low enough to enable a reinforcement learning algorithm to

quickly find successful ProMPs and high enough to express the diversity of trajectories that satisfy the criteria of our tasks, such as avoiding obstacles and reaching desired targets. In the experiment described in Section 7.1, the number of degrees of freedom controlled by the user is $n = 3$, while, in Section 7.2, $n = 6$. See Section 3 for the definition of ProMPs.

In the experiment described in Section 7.1, $p_{\text{progress}} = 0.8$ and $\Delta\nu = 0$, which means that, in that experiment, although the user was constrained by the learned guiding ProMPs, there was no force pulling him/her towards any of the targets. This made it convenient for the user to pause the movement while controlling the real robot. In Section 7.2, $p_{\text{progress}} = 0.8$ and $\Delta\nu = 0.5$, i.e., there was a small force pulling the user towards the target. See Section 5.1 for the explanation of the parameters p_{progress} and $\Delta\nu$. In both experiments described in this section, $p_{\text{switch}} = 1\mathrm{e}{-}20$ (see Section 5.2).

7.1. Shared Control of a Robot Arm

Picking up small objects through teleoperation can be a very challenging task for a human operator due to the difficulty in accurately estimating the 3D position of the objects of interest. The purpose of this experiment is to verify if our framework can learn multiple virtual guides to help a user pick up small objects by teleoperating a robot arm while avoiding obstacles. This experiment also tests if helpful virtual guides can be learned online in the face of changes in the environment, such as changes in the positions of the target objects and of the obstacle.

In our setup, the human manipulates a 6 DoF haptic device [18] to teleoperate a 7 DoF robot arm, the LBR iiwa 14 R820 manufactured by KUKA [19]. The human has control over the 3D Cartesian end-effector position of the robot arm and the joint configuration of the robot is computed through inverse kinematics. The goal of the human is to pick three little balls and release them inside a basket. The environment may change online through the introduction of an obstacle (a Pringles can). Our framework helps the user solve this task by learning multiple plans (ProMPs) that modulate the force feedback applied by the haptic device on the human. The ProMPs are relearned online whenever there are changes in the environment or when the belief of the freelance plan is larger than 0.5. Fig. 6 shows three snapshots during one trial to accomplish the pick and place task. The snapshots show that the virtual guides are learned online when the environment changes and that they are in accordance with the objectives of the task, which are reaching each of the balls to put them in the red basket and avoiding the obstacle (a Pringles can). The balls and the obstacle have been tracked by using the OptiTrack Flex 13 system [20].

In these experiments, the reward $r(\mathbf{w}) = \boldsymbol{\psi}(\mathbf{x_w})^\top \boldsymbol{\theta}$ is a weighted sum of features. The vector $\boldsymbol{\theta}$ has empirically determined weights for each feature in $\boldsymbol{\psi}(\mathbf{x_w})$. We have manually determined these weights by verifying if the distributions learned in a virtual model of our environment generated successful trajectories. An alternative to empirically determining the weights would be using inverse reinforcement learning (IRL) [21], which shall be investigated in our future research.

In this experiment, $\mathbf{x_w}(\nu_i) = [x(\nu_i), y(\nu_i), z(\nu_i)]^\top$ represents the 3D Cartesian position of the end-effector of the slave robot arm at phase ν_i. Our feature vector evaluated for trajectory $\mathbf{x}$ is
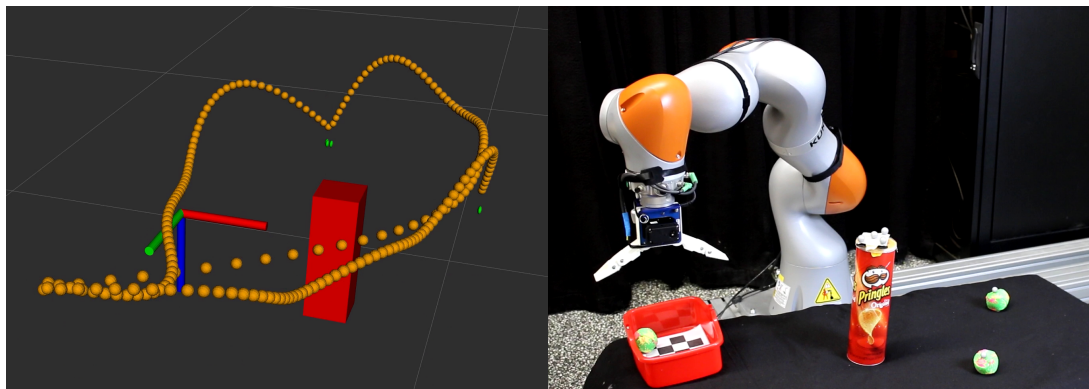
$$\boldsymbol{\psi}(\mathbf{x}) = \left[d_{\text{start}}^2, d_{\text{end}}^2, \mathcal{L}_{\text{box}}, \mathcal{L}_{\text{obstacle}}, \sum_{i=1}^{T_o} \dot{\mathbf{x}}^2(\nu_i), \sum_{i=1}^{T_o} \ddot{\mathbf{x}}^2(\nu_i), \sum_{i=1}^{T_o} \min(0.5, z(\nu_i)) \right]^\top \quad (17)$$

12
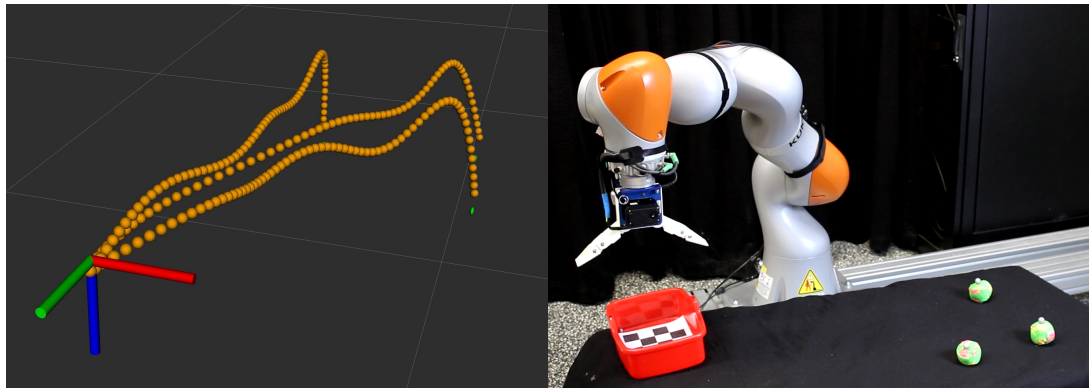
(a) Initial virtual guides (left) and environment (right)
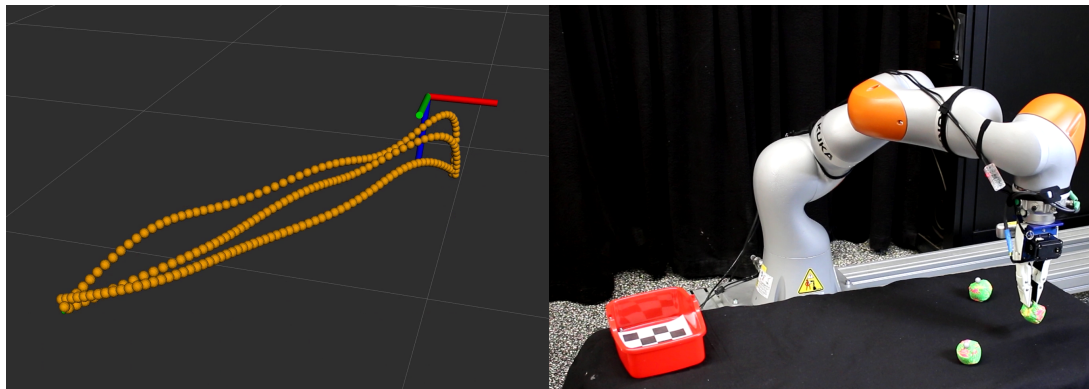
(b) Virtual guides (left) and environment (right) after first grasp

(c) Virtual guides (left) and environment (right) after introducing an obstacle

Figure 6. Teleoperating a robot arm through a haptic device with force feedback. In this experiment, the user controls the Cartesian position of the robot end-effector. The objective of the user is to put all the green balls inside the red basket while avoiding the obstacle (a Pringles can). Our framework computes distributions of trajectories that serve as virtual guides to the user. The force feedback applied by the haptic device on the user depends on the gradient of the trajectory distributions. Our framework replans the virtual guides whenever there is a change in the environment or when the user escapes all previously planned virtual guides.

and the vector of feature weights is $\boldsymbol{\theta} = [-5000, -5000, 5000, 5000, -500, -50000, 50]^\top$.

In (17), d_{start} is the distance between the start position of the trajectory and the desired start position. The variable d_{end} represents the distance between the end position and its closest target. The features $\mathcal{L}_{\text{box}}$ and $\mathcal{L}_{\text{obstacle}}$ prevent leaving the workspace in the form of a bounding box and collisions with the obstacle, respectively. The next two features prevent high velocities and accelerations. Finally, the last feature gives higher rewards for achieving high positions inside the bounding box defining the workspace of the robot. This feature induces grasping motions from the top.

The features $\mathcal{L}_{\text{box}}$ and $\mathcal{L}_{\text{obstacle}}$ are computed based on the signed Euclidean distance between the robot end-effector and the bounding box or obstacle, respectively. First, the minimum signed Euclidean distance d between trajectory $\mathbf{x}$ and the bounding box or obstacle is computed. Subsequently, the log-likelihood of that trajectory is given by

$$\mathcal{L} = \begin{cases} \log\left(\mathcal{N}\left(0; 0, \sigma^2\right)\right), \text{ if } d \geq 0 \\ \log\left(\mathcal{N}\left(d; 0, \sigma^2\right)\right), \text{ if } d < 0 \end{cases} \tag{18}$$

where we chose $\sigma^2 = 2$. This choice was based on our experience in simulation. By $\mathcal{N}\left(0; 0, \sigma^2\right)$ and $\mathcal{N}\left(d; 0, \sigma^2\right)$, we denote the probability density function of a univariate Gaussian with mean 0 and standard deviation σ evaluated at locations 0 and d, respectively. The signed Euclidean distance to the bounding box is positive inside the box and negative outside it. The obstacle is modeled as a cylinder. The signed Euclidean distance to the obstacle is positive outside the cylinder and negative inside it.

7.2. Shared Control of a Virtual Object

This experiment addresses assisting teleoperation with multiple solutions involving both translations and rotations. The main purpose of this experiment is to statistically test whether or not, in comparison with not using force feedback, our framework helps users avoid collisions and complete teleoperation tasks faster while retaining control over the haptic device. Here, users control a virtual pole, which allows for a simpler setup to perform user studies and minimizes the risk of damaging real equipment. Fig. 7 depicts our virtual environment. As in the previous experiment, the reward function has the form $r(\mathbf{w}) = \boldsymbol{\psi}(\mathbf{x_w})^\top \boldsymbol{\theta}$. This time, the feature vector for trajectory $\mathbf{x}$ is

$$\boldsymbol{\psi}(\mathbf{x}) = \left[d_{\text{start}}^2, d_{\text{end}}^2, \mathcal{L}_{\text{wall}}, \sum_{i=1}^{T_o} \dot{\mathbf{x}}^2(\nu_i), \sum_{i=1}^{T_o} \ddot{\mathbf{x}}^2(\nu_i), \sum_{i=1}^{T_o} \alpha^2(\nu_i) + \beta^2(\nu_i) + \gamma^2(\nu_i)\right]^\top \tag{19}$$

and the vector of feature weights is $\boldsymbol{\theta} = [-2.5, -5, 1000, -5, -5, -5]^\top$. As in Section 7.1, these values have been chosen based on our observations in simulations without the human of what weight combinations lead to the learning of successful trajectory distributions.

In (19), d_{start} is the distance between the start pose of the trajectory and the desired start pose. The variable d_{end} represents the distance between the end pose and the desired end pose. The feature $\mathcal{L}_{\text{wall}}$ prevents collisions with the obstacle (wall). The next two features prevent high velocities and accelerations, both Cartesian and rotational. Finally, the last feature prevents unnecessary rotations by punishing high Euler angles α, β and γ. The feature $\mathcal{L}_{\text{wall}}$ is computed according to (18) with d the

minimum signed Euclidean distance between the pole and the wall.

Ten users have been requested to perform two tasks as quickly as possible while avoiding collisions. Task 1 consisted of moving the pole from the start position to the end position through any of the two windows. Task 2 consisted of moving the pole from the start position to the end position through any of the two windows and bringing it back to the start position through the other window. The length of the pole is $2m$, the windows are squares with sides along x and z of $2m$, the wall is a square with sides along x and z of $100m$ and width (along y) of $3m$. The center of the wall is at $(x = 5m, y = 0m, z = -4m)$. The start position is at $(x = 10m, y = -30m, z = -5m)$ and the end position at $(x = 4m, y = 20m, z = -5m)$.

Task 1 has been performed by the users with two control modes: with force feedback or without force feedback. Task 2 has been performed with three control modes: with force feedback and replanning, with force feedback and without replanning, without force feedback. The order of the control modes experienced by each user has been determined at random. When force feedback was active, our system would compute a guiding ProMP. The initial ProMP resembles a tube starting at the start position and extending through one of the two windows until the end position. By applying a certain amount of force to the handle of the haptic device, users could escape the guidance of the initially planned ProMP. With replanning enabled, a new ProMP from the pole's current pose to the end pose would be planned once the user had escaped all the previously planned ProMPs. Without replanning, no new ProMP would be planned. Fig. 8 shows an example of replanning enabled.

In summary, each user performed Task 1 (start to end) two times (with and without force feedback) and Task 2 (round trip) three times (with force feedback and replanning, with force feedback and no replanning, without force feedback). When executing Task 2 with force feedback and no replanning, the users have been asked to skip the ProMP planned from start to end position, pass through one of the windows without force feedback and use the originally planned ProMP as a guide only when returning to the start position. The number of collisions with the wall in each trial has been recorded as well as the time users took to complete each task. Task 1 was considered completed once the distance between the center of the pole and the end position (marked by the blue sphere) was under $4m$. Task 2 was completed once, after reaching the position marked by the blue sphere, the distance between the center of the pole and the start position (marked by the yellow sphere) was under $4m$. Moreover, after all trials, users have been asked "On a scale from 1 to 5, how much did you feel in control of the haptic device? 1 means completely not in control. 5 means completely in control." Fig. 9 shows the results.
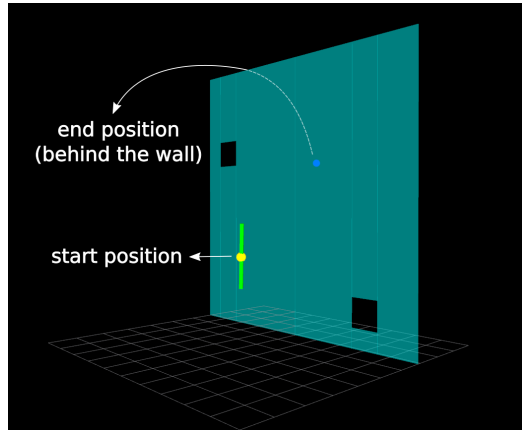

Figure 7. 6D Experiment. The user operates the haptic device Haption Virtuose 6D TAO to translate and rotate a pole in a virtual environment. The goal of the user is to move the green pole from the start position (yellow sphere) to the end position (blue sphere) through one of the two windows without hitting the wall. VIPS learns a mixture of trajectory distributions (ProMPs) that helps the user achieve the necessary translations and rotations to solve this task.
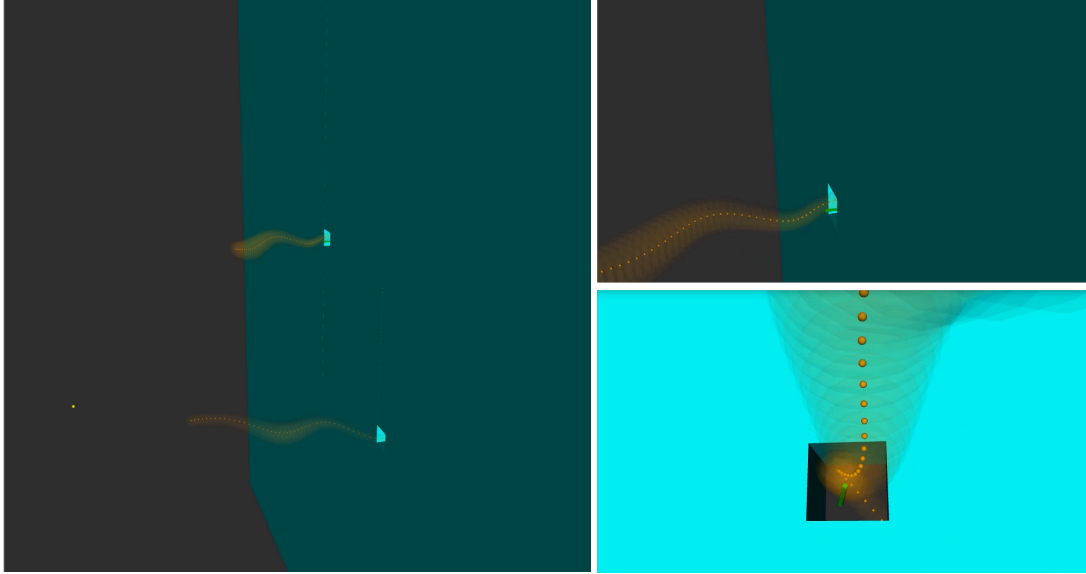
Figure 8. Example of replanning in the experiment involving the teleoperation of a virtual pole through a haptic device. The little orange spheres represent positions along the mean of distributions of trajectories (ProMPs). The larger transparent orange spheres represent the variances in x, y and z corresponding to the positions depicted by the small spheres. Our framework learns ProMPs to help the user move the pole through the windows to a target position on the other side of the wall. When the user escapes the attraction of one of the ProMPs, another ProMP is learned to take the pole from its current position to the target position. The three images represent the same moment from three different perspectives.

Nonparametric ANOVA Kruskal-Wallis hypothesis tests with a significance level $\alpha = 0.05$ have been conducted. For the evaluations involving more than two groups, post-hoc Conover's tests have been used to indicate which groups were significantly different.

There was a significant difference between "with force feedback" and "no force feedback" with respect to the number of collisions ($p = 0.0137$) and time to complete the task ($p = 0.0002$) in Task 1. The difference between 'with force feedback" and "no force feedback" with respect to the feeling of control reported by the users was not significant ($p = 0.9060$) in Task 1.

Kruskal-Wallis test has indicated a significant difference ($p = 0.0008$) between the three teleoperation modes in Task 2 with respect to the number of collisions. Conover's test has indicated that the differences were significant between "no force feedback" and "with force feedback, no replanning" ($p = 0.0035$) as well as between "no force feedback" and "with force feedback, with replanning" ($p = 0.0001$). Kruskal-Wallis test has indicated no significant difference ($p = 0.2050$) between the teleoperation modes in Task 2 with respect to the time users took to complete the task. There was also no significant difference ($p = 0.7563$) between the three teleoperation modes in Task 2 with respect to the feeling of control reported by the users.

Based on the box plots and hypothesis tests, we conclude that our framework helps users avoid collisions in both tasks. Furthermore, our framework helps users complete Task 1 faster than without assistance. Although the hypothesis tests did not indicate a significant difference between the teleoperation modes in Task 2 with respect to the time users took to complete the task, Fig. 9(b) shows two outliers for the teleoperation mode without force feedback. The modes with force feedback did not present outliers.

16

Therefore, our framework may help users to not require a very large amount of time to complete Task 2. Moreover, the box plots and hypothesis tests indicate that our framework helps users avoid collisions in Task 2 without compromising the time users take to complete the task.

The hypothesis tests indicate that the difference between the medians of the feeling of control reported by the users was not significant in any of the tasks. We conclude that our framework does not affect much the feeling of control of the users over the haptic device. This result is in accordance with the intended purposes of our framework, since it is supposed to assist users while giving them the freedom to diverge from the guidance of the haptic device.

8. Conclusion and Future Work

This paper introduced a new approach to assisted teleoperation. A GMM of trajectories is learned through variational inference with the VIPS algorithm. VIPS learns a GMM that maximizes an episodic reward function in expectation while maintaining high entropy. Each mixture component is a ProMP. The gradient of the logarithm of the marginal distribution in end-effector space plus a damping term is equal to the wrench applied by the haptic device on the user. Moreover, Bayesian belief update is used to infer the plan and the phase intended by the user. The beliefs about the plan and the phase correspond to the weights of the mixture components in end-effector space. Therefore, these beliefs influence the wrench applied by the haptic device on the user.

Our approach can guide the user around obstacles when classical potential fields methods could get stuck in local minima. It can learn multimodal solutions to a task and adapt online to changes in the environment or in the intention of the user, as it has been demonstrated in our experiments. User studies have demonstrated that our framework can help users perform teleoperation tasks faster and avoiding collisions with obsta-
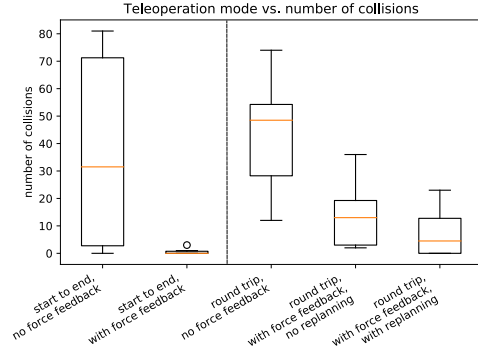
(a) Teleoperation mode versus number of collisions
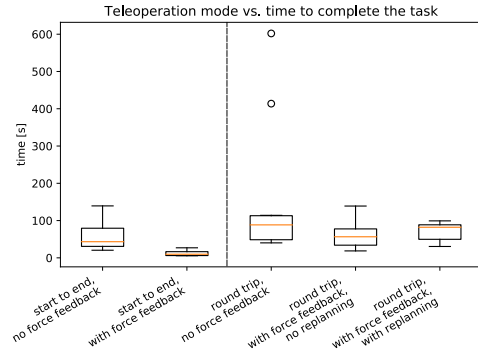
(b) Teleoperation mode versus time to complete the task
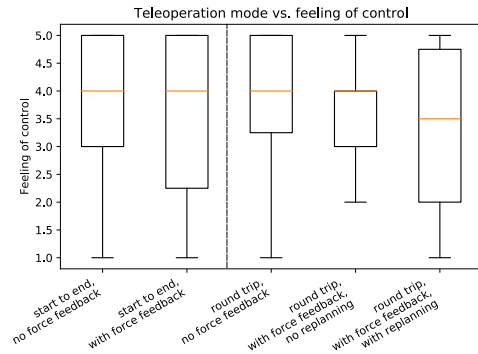
(c) Teleoperation mode versus feeling of control

Figure 9. Box plots representing the data acquired in our user studies on the control of the 6D pose of a virtual pole by manipulating a haptic device.

17

cles.

A promising line of research for future work would be to combine the proposed framework with inverse reinforcement learning to avoid specifying the weights of the features of the reward function manually. In addition, user studies involving the teleoperation of a real robot arm would be interesting to evaluate the efficacy of our approach in more realistic scenarios. In this case, our system should also give the user haptic cues to avoid singularities and joint limits.

Funding

References

[1] Bloss R. How do you decommission a nuclear installation? call in the robots. Industrial Robot: An International Journal. 2010;37(2):133–136.

[2] Havoutis I, Calinon S. Learning from demonstration for semi-autonomous teleoperation. Autonomous Robots. 2019;43(3):713–726.

[3] Abi-Farraj F, Pacchierotti C, Arenz O, et al. A haptic shared-control architecture for guided multi-target robotic grasping. In: IEEE Transactions on Haptics (ToH); 2019.

[4] Ewerton M, Rother D, Weimar J, et al. Assisting movement training and execution with visual and haptic feedback. Frontiers in neurorobotics. 2018;12:24.

[5] Paraschos A, Daniel C, Peters J, et al. Probabilistic movement primitives. In: Advances in Neural Information Processing Systems (NIPS); 2013. p. 2616–2624.

[6] Arenz O, Zhong M, Neumann G. Efficient gradient-free variational inference using policy search. In: International Conference on Machine Learning; 2018. p. 234–243.

[7] Aigner P, McCarragher B. Human integration into robot control utilising potential fields. In: Proceedings of International Conference on Robotics and Automation; Vol. 1; IEEE; 1997. p. 291–296.

[8] Howard AM, Park CH. Haptically guided teleoperation for learning manipulation tasks. In: Proceedings of the Robotics, Science & Systems: Workshop on Manipulation for Human Environments; 2007.

[9] Gioioso G, Mohammadi M, Franchi A, et al. A force-based bilateral teleoperation framework for aerial robots in contact with the environment. In: 2015 IEEE International Conference on Robotics and Automation (ICRA); IEEE; 2015. p. 318–324.

[10] Huber L, Billard A, Slotine JJ. Avoidance of convex and concave obstacles with convergence ensured through contraction. IEEE Robotics and Automation Letters. 2019; 4(2):1462–1469.

[11] Khansari-Zadeh SM, Khatib O. Learning potential functions from human demonstrations with encapsulated dynamic and compliant behaviors. Autonomous Robots. 2017;41(1):45–69.

[12] Ewerton M, Neumann G, Lioutikov R, et al. Learning multiple collaborative tasks with a mixture of interaction primitives. In: 2015 IEEE International Conference on Robotics and Automation (ICRA); IEEE; 2015. p. 1535–1542.

[13] Koert D, Trick S, Ewerton M, et al. Online learning of an open-ended skill library for collaborative tasks. In: 2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids); IEEE; 2018. p. 1–9.

[14] Engel PM, Heinen MR. Incremental learning of multivariate gaussian mixture models. In: Brazilian Symposium on Artificial Intelligence; Springer; 2010. p. 82–91.

[15] Raiola G, Lamy X, Stulp F. Co-manipulation with multiple probabilistic virtual guides. In: Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on; IEEE; 2015. p. 7–13.

[16] Pignat E, Lembono T, Calinon S. Variational inference with mixture model approximation: Robotic applications. In: arXiv preprint arXiv:1905.09597; 2019.

[17] Bishop CM. Pattern recognition and machine learning. Springer; 2006.

[18] Haption. Virtuose 6d tao specifications [https://www.haption.com/en/products-en/virtuose-6d-tao-en.html]; 2020.

[19] KUKA. Lbr iiwa specifications [https://www.kuka.com/en-ch/products/robotics-systems/industrial-robots/lbr-iiwa]; 2020.

[20] OptiTrack. Optitrack flex 13 specifications [https://optitrack.com/products/flex-13/]; 2020.

[21] Abbeel P, Ng AY. Apprenticeship learning via inverse reinforcement learning. In: Proceedings of the twenty-first International Conference on Machine Learning; ACM; 2004. p. 1–8.

Appendix A. Definition of the Shifting Operator

The shifting operator $\mathcal{T}_{\Delta\nu}(p)$ is defined as a function that maps a discrete probability distribution p over T_o phases to a new distribution over phases by shifting its probability mass by $\Delta\nu$ into the future (see Section 5.1). Formally, $\mathcal{T}_{\Delta\nu}$ is a function

$$\mathcal{T}_{\Delta\nu} : \mathcal{S}^{T_o-1} \mapsto \mathcal{S}^{T_o-1}, \tag{A1}$$

where $\mathcal{S}^n$ denotes the n-simplex. The ith vertex of the shifted simplex $\mathcal{T}_{\Delta\nu}(p)^{(i)}$ is defined as

$$\mathcal{T}_{\Delta\nu}(p)^{(i)} = \begin{cases} \mathbb{1}_{i-\lfloor\Delta\nu\rfloor>1}\left[(1 - \mathbb{1}_{\Delta\nu<1}[(\Delta\nu - \lfloor\Delta\nu\rfloor)])p^{(i-\lfloor\Delta\nu\rfloor)}\right] \\ \qquad\qquad + \mathbb{1}_{i-\lceil\Delta\nu\rceil>1}\left[(\Delta\nu - \lfloor\Delta\nu\rfloor)p^{(i-\lceil\Delta\nu\rceil)}\right] &, \quad \text{if } i < T_o \\ \sum_{i=1}^{T_o-1}\mathcal{T}_{\Delta\nu}(p)^{(i)}, & \text{if } i = T_o. \end{cases} \tag{A2}$$