A Generative Model for Intention Recognition and Manipulation Assistance in Teleoperation

Ajay Kumar Tanwani[1,2], Sylvain Calinon[1]

***Abstract*— Performing remote manipulation tasks by teleoperation with limited bandwidth, communication delays and environmental differences is a challenging problem. In this paper, we learn a task-parameterized generative model from the teleoperator demonstrations using a *hidden semi-Markov model* that provides assistance in performing remote manipulation tasks. We present a probabilistic formulation to capture the intention of the teleoperator, and subsequently assist the teleoperator by time-independent shared control and/or time-dependent autonomous control formulations of the model. In the shared control mode, the model corrects the remote arm movement based on the current state of the teleoperator; whereas in the autonomous control mode, the model generates the movement of the remote arm for autonomous task execution. We show the formulation of the model with *virtual fixtures* and provide comparisons to benchmark our approach. Teleoperation experiments with the Baxter robot for reaching a movable target and opening a valve reveal that the proposed methodology improves the performance of the teleoperator and caters for environmental differences in performing remote manipulation tasks.**

I. Introduction

Teleoperated robots are going to increasingly assist humans in performing everyday life tasks as diverse as minimally invasive surgeries, security/surveillance, telepresence, warehouse management, remote patient monitoring, inspection/exploration in deep underwater or space missions. Teleoperation provides a low cost solution to offload tedious work from humans and reach distant and/or hazardous environments. Advancing the state-of-the-art in teleoperation is the central focus of many research programs, including DARPA Robotics and NASA Space Robotics challenges. Improving autonomy in teleoperation, however, poses all kind of challenges to the existing techniques due to limited bandwidth, communication latency, and environmental differences between the teleoperator and the remote sites.

Within the DexROV project, we aim at achieving dexterous manipulation tasks in remote underwater environments [1]. Large communication delays with satellite communication render direct teleoperation infeasible, thereby, requiring semi-autonomous capabilities of the remotely operated vehicle to carry out manipulation tasks. Moreover, the operational costs are significantly reduced by moving the teleoperation personnel from the vessel to operate the vehicle from a remote facility. We use the two-armed Baxter robot as a mock-up of the teleoperation system, i.e., one arm becomes the input device for the teleoperator, and the other one is used for performing the manipulation task. The operator controls/teleoperates the remote arm with a simulated delay using the other arm by getting visual feedback from the remote arm. A set of kinesthetic demonstrations of the teleoperator is used to teach the robot how to perform each task. We seek to leverage upon probabilistic generative models to detect the intention of the teleoperator, and assist the movement on the robot side under varying environmental situations.

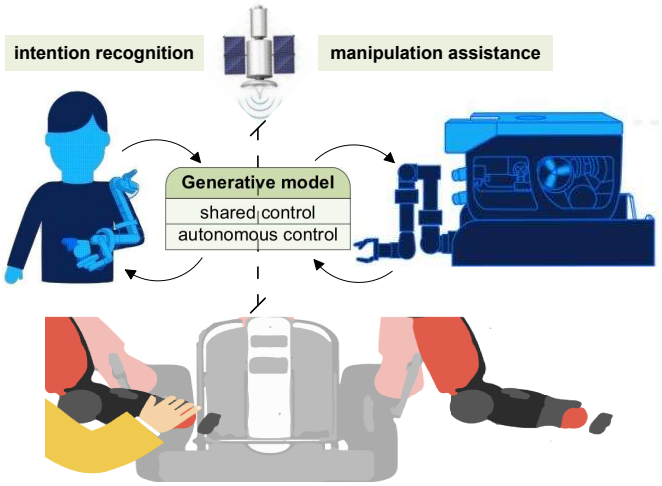

Fig. 1: *(top)* generative model locally recognizes the intent of the teleoperator and provides manipulation assistance, *(bottom)* teleoperation mock-up with the two-armed Baxter robot where one arm is used as input device for the teleoperator and the other arm is used to perform remote manipulation tasks.

In this paper, we use a task-parameterized *hidden semi-Markov model* (HSMM) [2], [3], [4] for semi-autonomous teleoperation of remote manipulation tasks. The HSMM clusters the demonstrations into meaningful segments or primitives and encodes the transition patterns among the segments. We assist the teleoperator using the learned model by: 1) continuously correcting the movement of the remote arm given the teleoperator arm data based on shared control, or 2) generating the movement of the remote arm based on autonomous control (see Fig. 1). We also show the formulation of the model with virtual fixtures to benchmark our approach. The predicted movement from the model is smoothly followed by the remote arm with a discrete-time linear quadratic tracker/regulator. Task-parameterized

[1]Idiap Research Institute, Switzerland. {ajay.tanwani, sylvain.calinon}@idiap.ch
[2]Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland.
This work was in part supported by the DexROV project through the EC Horizon 2020 programme (Grant #635491).

formulation of the model allows the teleoperator to perform the task with respect to the perceived environment (often delayed/inconsistent compared to the actual remote environment), and the remote robot arm to adapt the movement locally in accordance with the actual situation. Our experiments reveal that the proposed formulations of the model improves the performance of the teleoperator by mitigating the effect of imprecise movements.

A. Background and Related Work

Teleoperated robots are traditionally based on master-slave architecture where the teleoperator (master) transmits position/force to the robot (slave) in a unilateral mode, or transmits and receives position/force via bilateral communication (see [5] for a detailed review). Bilateral teleoperation uses a haptic interface to make the operator feel a particular impedance relative to the slave position or the force recorded between the slave and the environment. Despite the simple mechanism, teleoperation requires skilled personnel to remotely operate the robot, while having limited access to the controllable degrees of freedom and the sensory feedback. Moreover, stability issues arise in handling environmental uncertainty with communication delays between the teleoperator and the robot. This has motivated several control theoretic solutions such as scattering approach, wave variables, passivity based control, multichannel feedback and model prediction based control to deal with delayed force reflections [6]. Modern day teleoperation systems use additional interfaces such as exoskeleton and/or head mounted display to increase the sense of *telepresence* in performing the task [7].

The teleoperator controls the remote robot using either: 1) direct control, 2) shared control, or 3) supervisory control. Direct teleoperation lacks the autonomy/intelligence to assist the operator and the remote robot simply mimics the movement of the teleoperator. Shared control corrects/fine-tunes the continuously streamed teleoperator data by local sensory feedback on the remote side. For constrained manipulation tasks, *virtual fixtures* have been used to reduce the operator workload by influencing the robot motion along desired paths [8], [9]. Supervisory or autonomous control gives local autonomy to the remote robot to execute manipulation tasks in the presence of large communication delays. It makes use of predictive displays and high-level symbolic commands of atomic structure (such as reach, grasp, etc.) to breakdown a task in smaller subtasks [10], [11].

Robot learning from demonstrations is a promising approach to assist humans in performing daily life tasks (see [12], [13] for an overview). In this context, advancing autonomy in teleoperation addresses two main problems: 1) predicting the operator's intent while performing the task, and 2) deciding how to assist the teleoperator. Both aspects are closely related in cooperative robots for human-robot collaboration [14], and in general describe the *what-to-imitate* and *how-to-imitate* problems in programming by demonstration. Depending upon how the word *intention* is phrased, a vast amount of literature exists to encapsulate the behaviour of the operator, and subsequently decode it for assistance. For example, predicting the user intent can be posed as a classification problem of reaching a particular goal position in a predefined set of goals [15]. Alternatively, the user may be assumed to maximize an unknown reward function to be ascertained by inverse reinforcement learning (IRL). Dragan and Srinivasa formulated a policy blending mechanism to combine the teleoperator intention with the robot movement using IRL [16]. In cognitive science, Bayesian models are more commonly used to incorporate uncertainty in decoding the user behaviour. Hauser in [17] inferred the type of task performed by the user with a Bayesian Gaussian mixture auto-regression framework, and followed the predicted trajectory with a cooperative motion planner.

Generative models such as Hidden Markov Models (HMMs) have been widely used to interpret human intention as performing a discrete set of tasks/subtasks with common low level sensory observations. The use of hierarchical representations [18], or sets of dynamic models for the subtasks sequenced together with a Markov chain [19], have been investigated to describe several human behaviours. Li *et al.* used virtual fixtures with an HMM to segment if the user intends to follow a periodic motion curve, not follow the curve or stay idle [20]. Nolin *et al.* in [21] investigated settings of discrete compliance levels with an HMM, namely $\{toggle, fade, hold\}$, to assist the user in following the virtual fixture based on his/her demonstration. Roila *et al.* presented probabilistic virtually guided fixtures for assistance [22]. Medina *et al.* perform task segmentation with an HMM and incrementally update its parameters during reproduction to progressively increase the collaborative role of the robot in performing the task [23]. Wang *et al.* infer the probability distribution over intentions from the human observations in the latent state of a Gaussian process dynamical model [24]. Maeda *et al.* recognize the phase/stage of human movement from intermittent observations under different possible speeds and plan a collaborative trajectory for the robot [25].

This paper extends our previous work on encoding manipulation skills in a task-adaptive manner [4], [26], [27] to the context of semi-autonomous teleoperation. Our major contributions are the probabilistic formulations to assist the user by correcting the robot motion with shared control, synthesizing the motion with autonomous control, and following a desired trajectory with virtual fixtures control.

II. Semi-Autonomous Teleoperation

In this section, we first present an example to illustrate the semi-autonomous teleoperation scenario with the Baxter robot. We then describe our learning approach for encoding the task from teleoperator demonstrations, followed by control formulations to assist the teleoperator in performing manipulation tasks.

A. Illustrative Example

Consider the task of grasping an object on the remote site by teleoperation. The task is demonstrated on the teleoperator

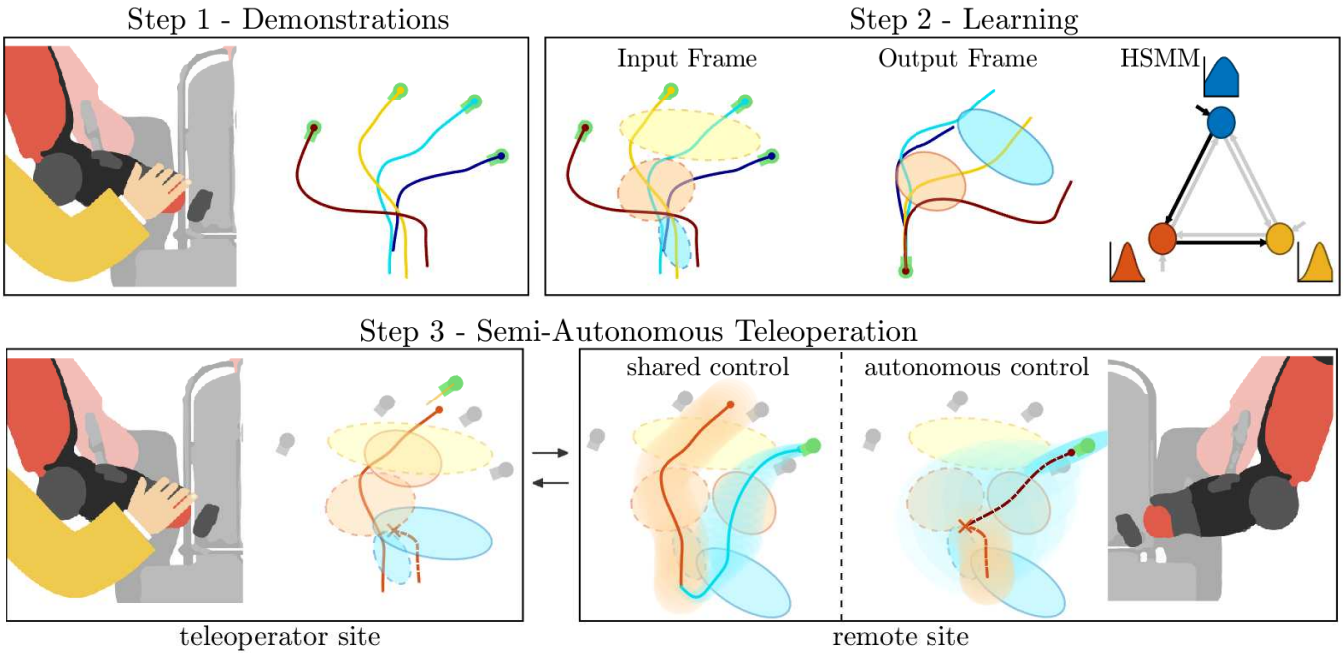

Fig. 2: Semi-autonomous teleoperation framework: Step 1) the teleoperator provides a few demonstrations of the manipulation task under different object positions shown in green (the points depict the end of the demonstrations); Step 2) a task-parameterized HSMM is learned, with the input reference frame representing the demonstrations in the global coordinate system, and the output reference frame representing the demonstrations in the coordinate system attached to the object (the Gaussian depicted as an ellipse represents the emission distribution of a state, and the graphical representation of the HSMM shows transition among states and the state duration modeled with a Gaussian); Step 3) *(left)* the teleoperator performs an imprecise movement (in orange) to grasp the perceived object in green, *(right)* the shared control mode locally corrects the movement of the robot (in blue) in accordance with the actual object position on the remote site, while the autonomous control mode generates the movement to the object (in dark red) after the teleoperator switches to the autonomous mode (marked with a cross). Note that the output frame component adapts the model locally in accordance with the object.

site from different initial configurations of the arm and the object. After learning the model from a few demonstrations, the model parameters are passed to the remote site during the start of the mission (implemented as a ROS service). During teleoperation, the teleoperator arm data is continuously streamed to the remote site, while the remote robot arm data and the object description (reference frames described as task parameters) are sent back to the teleoperator side. To simulate communication latency in teleoperation, data is buffered on both the teleoperation and the remote sites. Fixed time delays of up to 2 seconds are introduced, under which the teleoperator perceives the object with delayed feedback.

The teleoperator has two modes of assistance as illustrated in Fig. 2: 1) shared control, and 2) autonomous control. Shared control continuously adjusts/corrects the robot movement given the teleoperator arm data based on the learned model that locally adapts according to the object position. The model exploits the variability observed in the teleoperator demonstrations. Where the variance is high such as away from the object, the correction is mild, whereas for low variance regions close to the object, the model strongly corrects the remote arm to track the object. Supervisory control gives the teleoperator more autonomy as the model detects the state of the task and generates the remote arm movement to accomplish the task.

B. Learning Task-Parameterized HSMM

We use one arm of the Baxter robot as the input device for the teleoperator, with a controller compensating for the effect of gravity. Let us denote an observation of the teleoperator arm as $\boldsymbol{\xi}_t \in \mathbb{R}^D$ with $\boldsymbol{\xi}_t = \left[\boldsymbol{\xi}_t^{\mathcal{I}^\top} \ \boldsymbol{\xi}_t^{\mathcal{O}^\top}\right]^\top$ where $\boldsymbol{\xi}_t^{\mathcal{I}}$ and $\boldsymbol{\xi}_t^{\mathcal{O}}$ respectively represent the pose of the end-effector of the teleoperator arm at time t in a global frame of reference and the same pose observed with respect to another frame of reference describing the current context or situation (superscripts $\mathcal{I}$ and $\mathcal{O}$ represent the input and the output components). The aim of augmenting the teleoperator pose with different frames of reference is to couple the movement of the teleoperator arm with external environmental variables, i.e., we learn the mapping between the teleoperator pose in two reference frames: in a global frame and in the object frame, modeled as a joint distribution. We assume that the reference frames are specified by the user, based on prior knowledge about the carried out task. Typically, reference frames will be attached to objects, tools or locations that could be relevant in the execution of a task.

The pose of the teleoperator arm describes the position $\boldsymbol{x}_t^p \in \mathbb{R}^3$ and the unit quaternion orientation $\boldsymbol{\varepsilon}_t^o \in \boldsymbol{S}^3$

of the end-effector ($D = 14$ with 7 dimensional pose of the input dimension and 7 dimensional pose of the output dimension observed with respect to task parameters). We represent the task parameters with P coordinate systems, defined by the reference frames $\{\boldsymbol{A}_j, \boldsymbol{b}_j\}_{j=1}^{P}$, where $\boldsymbol{A}_j$ denotes the orientation of the frame and $\boldsymbol{b}_j$ represents the origin of the frame, described by

$$\boldsymbol{A}_j = \begin{bmatrix} \boldsymbol{I}^{\mathcal{I}} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}_j^{\mathcal{O}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \mathcal{E}_j^{\mathcal{O}} \end{bmatrix}, \boldsymbol{b}_j = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{p}_j^{\mathcal{O}} \\ \boldsymbol{0} \end{bmatrix}, \tag{1}$$

where $\boldsymbol{p}_j^{\mathcal{O}} \in \mathbb{R}^3$, $\boldsymbol{R}_j^{\mathcal{O}} \in \mathbb{R}^{3\times3}$, $\mathcal{E}_j^{\mathcal{O}} \in \mathbb{R}^{4\times4}$ denote the Cartesian position, the rotation matrix and the quaternion matrix of the j-th frame, respectively.

The observation sequence $\{\boldsymbol{\xi}_t\}_{t=1}^{T}$ of T datapoints, observed from the perspective of different reference frames, forms a third order tensor dataset $\{\boldsymbol{\xi}_t^{(j)}\}_{t,j=1}^{T,P}$ with $\boldsymbol{\xi}_t^{(j)} = \boldsymbol{A}_j^{-1}(\boldsymbol{\xi}_t - \boldsymbol{b}_j)$. This dataset is used to train a task-parameterized HSMM with K hidden states represented by the parameter set $\theta_h = \left\{\Pi_i, \{a_{i,m}\}_{m=1}^{K}, \{\boldsymbol{\mu}_i^{(j)}, \boldsymbol{\Sigma}_i^{(j)}\}_{j=1}^{P}, \mu_i^S, \Sigma_i^S\right\}_{i=1}^{K}$. Π_i denotes the initial state distribution, $\boldsymbol{a} \in \mathbb{R}^{K\times K}$ with $a_{i,j} \triangleq P(j|i)$ denote the transition probability of moving from state i at time $t-1$ to state j at time t, the output distribution of state i in frame j is described by a multivariate Gaussian with parameters $\{\boldsymbol{\mu}_i^{(j)}, \boldsymbol{\Sigma}_i^{(j)}\}$, and the parameters $\{\mu_i^S, \Sigma_i^S\}$ represent the mean and the standard deviation of staying s consecutive steps in state i estimated by a Gaussian $\mathcal{N}(s|\mu_i^S, \Sigma_i^S)$. The K Gaussian components for each reference frame constitute a Gaussian mixture model with a shared transition matrix and state duration model to capture the sequential patterns in the demonstrations. The parameters $\left\{\Pi_i, \{a_{i,m}\}_{m=1}^{K}, \{\boldsymbol{\mu}_i^{(j)}, \boldsymbol{\Sigma}_i^{(j)}\}_{j=1}^{P}\right\}_{i=1}^{K}$ are estimated using an expectation maximization (EM) algorithm [2] with latent space parameter updates, while the parameters $\{\mu_i^S, \Sigma_i^S\}_{i=1}^{K}$ are estimated by fitting a duration model on each state from the computed hidden state sequence. HSMM, in comparison to HMM, better models movements with longer state dwell times, see [28] for comparison with other mixture models.

C. Model Adaptation in New Situations

For a given environmental situation represented by the reference frames $\{\tilde{\boldsymbol{A}}_j, \tilde{\boldsymbol{b}}_j\}_{j=1}^{P}$, the resulting model parameters $\{\tilde{\boldsymbol{\mu}}_i, \tilde{\boldsymbol{\Sigma}}_i\}$ are obtained by first linearly transforming the Gaussians in the P reference frames with

$$\mathcal{N}(\tilde{\boldsymbol{\mu}}_i^{(j)}, \tilde{\boldsymbol{\Sigma}}_i^{(j)}) = \mathcal{N}\left(\tilde{\boldsymbol{A}}_j \boldsymbol{\mu}_i^{(j)} + \tilde{\boldsymbol{b}}_j, \tilde{\boldsymbol{A}}_j \boldsymbol{\Sigma}_i^{(j)} \tilde{\boldsymbol{A}}_j^{\top}\right), \tag{2}$$

and then computing the products of the linearly transformed Gaussians for each component [29], [4] with

$$\mathcal{N}(\tilde{\boldsymbol{\mu}}_i, \tilde{\boldsymbol{\Sigma}}_i) \propto \prod_{j=1}^{P} \mathcal{N}(\tilde{\boldsymbol{\mu}}_i^{(j)}, \tilde{\boldsymbol{\Sigma}}_i^{(j)}), \tag{3}$$

$$\tilde{\boldsymbol{\Sigma}}_i = \left(\sum_{j=1}^{P} \tilde{\boldsymbol{\Sigma}}_i^{(j)}\right)^{-1} \tilde{\boldsymbol{\mu}}_i = \tilde{\boldsymbol{\Sigma}}_i \sum_{j=1}^{P} \left(\tilde{\boldsymbol{\Sigma}}_i^{(j)}\right)^{-1} \left(\tilde{\boldsymbol{\mu}}_i^{(j)}\right).$$

D. Manipulation Assistance

We first present two formulations of the learned model to assist the teleoperator in performing remote manipulation tasks: 1) *time-independent shared control*, 2) *time-dependent autonomous control*, and then show comparison of our approach with virtual fixtures.

1) Shared Control: In shared control, we seek to continuously correct the movement of the robot arm according to the learned model given the streaming input data from the teleoperator. To this end, we first approximate the conditional probability distribution of the teleoperator pose in each output reference frame component given the current teleoperator pose as $\mathcal{P}(\boldsymbol{\xi}_t^{\mathcal{O}_j}|\boldsymbol{\xi}_t^{\mathcal{I}}) \approx \mathcal{N}(\tilde{\boldsymbol{\mu}}_t^{\mathcal{O}_j}, \tilde{\boldsymbol{\Sigma}}_t^{\mathcal{O}_j})$, based on the joint distribution of the linearly transformed Gaussians $\mathcal{N}(\tilde{\boldsymbol{\mu}}_i^{(j)}, \tilde{\boldsymbol{\Sigma}}_i^{(j)})$. Denoting the block decomposition of the joint distribution as

$$\tilde{\boldsymbol{\mu}}_i^{(j)} = \begin{bmatrix} \tilde{\boldsymbol{\mu}}_i^{\mathcal{I}_j} \\ \tilde{\boldsymbol{\mu}}_i^{\mathcal{O}_j} \end{bmatrix}, \quad \tilde{\boldsymbol{\Sigma}}_i^{(j)} = \begin{bmatrix} \tilde{\boldsymbol{\Sigma}}_i^{\mathcal{I}_j} & \tilde{\boldsymbol{\Sigma}}_i^{\mathcal{I}\mathcal{O}_j} \\ \tilde{\boldsymbol{\Sigma}}_i^{\mathcal{O}\mathcal{I}_j} & \tilde{\boldsymbol{\Sigma}}_i^{\mathcal{O}_j} \end{bmatrix}, \tag{4}$$

the conditional output distribution $\mathcal{N}(\tilde{\boldsymbol{\mu}}_t^{\mathcal{O}_j}, \tilde{\boldsymbol{\Sigma}}_t^{\mathcal{O}_j})$ is approximated using Gaussian mixture regression [30],

$$\tilde{\boldsymbol{\mu}}_t^{\mathcal{O}_j} = \sum_{i=1}^{K} h_i(\boldsymbol{\xi}_t^{\mathcal{I}}) \ \hat{\boldsymbol{\mu}}_i^{\mathcal{O}_j}(\boldsymbol{\xi}_t^{\mathcal{I}}), \tag{5}$$

$$\tilde{\boldsymbol{\Sigma}}_t^{\mathcal{O}_j} = \sum_{i=1}^{K} h_i(\boldsymbol{\xi}_t^{\mathcal{I}}) \left(\hat{\boldsymbol{\Sigma}}_i^{\mathcal{O}_j} + \hat{\boldsymbol{\mu}}_i^{\mathcal{O}_j}(\boldsymbol{\xi}_t^{\mathcal{I}}) \hat{\boldsymbol{\mu}}_i^{\mathcal{O}_j}(\boldsymbol{\xi}_t^{\mathcal{I}})^{\top}\right) - \tilde{\boldsymbol{\mu}}_t^{\mathcal{O}_j} \tilde{\boldsymbol{\mu}}_t^{\mathcal{O}_j\top}, \tag{6}$$

$$\text{with} \quad h_i(\boldsymbol{\xi}_t^{\mathcal{I}}) = \frac{\pi_i \, \mathcal{N}(\boldsymbol{\xi}_t^{\mathcal{I}}|\, \tilde{\boldsymbol{\mu}}_i^{\mathcal{I}_j}, \tilde{\boldsymbol{\Sigma}}_i^{\mathcal{I}_j})}{\sum_k^K \pi_k \, \mathcal{N}(\boldsymbol{\xi}_t^{\mathcal{I}}|\, \tilde{\boldsymbol{\mu}}_k^{\mathcal{I}_j}, \tilde{\boldsymbol{\Sigma}}_k^{\mathcal{I}_j})}, \tag{7}$$

$$\hat{\boldsymbol{\mu}}_i^{\mathcal{O}_j}(\boldsymbol{\xi}_t^{\mathcal{I}}) = \tilde{\boldsymbol{\mu}}_i^{\mathcal{O}_j} + \tilde{\boldsymbol{\Sigma}}_i^{\mathcal{O}\mathcal{I}_j} \tilde{\boldsymbol{\Sigma}}_i^{\mathcal{I}_j\,-1} (\boldsymbol{\xi}_t^{\mathcal{I}} - \tilde{\boldsymbol{\mu}}_i^{\mathcal{I}_j}), \tag{8}$$

$$\hat{\boldsymbol{\Sigma}}_i^{\mathcal{O}_j} = \tilde{\boldsymbol{\Sigma}}_i^{\mathcal{O}_j} - \tilde{\boldsymbol{\Sigma}}_i^{\mathcal{O}\mathcal{I}_j} \tilde{\boldsymbol{\Sigma}}_i^{\mathcal{I}_j\,-1} \tilde{\boldsymbol{\Sigma}}_i^{\mathcal{I}\mathcal{O}_j}. \tag{9}$$

The conditional probability distribution $\mathcal{N}(\tilde{\boldsymbol{\mu}}_t^{\mathcal{O}_j}, \tilde{\boldsymbol{\Sigma}}_t^{\mathcal{O}_j})$ predicts the teleoperator pose according to the learned model, and the uncertainty associated with the pose in the given frame $\{\tilde{\boldsymbol{A}}_j, \tilde{\boldsymbol{b}}_j\}$. The conditional probability distributions of all reference frames are fused using the product of Gaussians to yield the desired pose at each time instant, $\mathcal{N}(\hat{\boldsymbol{\mu}}_t, \hat{\boldsymbol{\Sigma}}_t) \propto \prod_{j=1}^{P} \mathcal{N}(\tilde{\boldsymbol{\mu}}_t^{\mathcal{O}_j}, \tilde{\boldsymbol{\Sigma}}_t^{\mathcal{O}_j})$ (see Eq. (II-C)). Note that the variance of the resulting product of Gaussians determines the trade-off between direct teleoperation and correction applied by the model. If the variance is low, the correction is strong and the robot arm follows the model better than the teleoperator. A similar variance based shared control architecture has also been adopted by authors in [31].

2) Autonomous Control: Continuously operating the remote arm for routine tasks can be cumbersome for the teleoperator, especially in the presence of communication latency. In such a situation, the teleoperator may switch at any point in time t_o to the autonomous control mode upon which the robot arm recursively re-plans and executes the task for the next T steps. When the task is accomplished or the communication channel is re-established, the operator

switches back to the direct/shared control upon which the robot arm returns to the desired teleoperated state (see [32] for application of this approach to adaptive dressing skills).

The input part of the learned model is used to recognize the most likely state of the task at t_o given the teleoperator pose $\boldsymbol{\xi}_t^{\mathcal{I}}$. The desired movement sequence is then computed with the help of the *forward variable*, $\alpha_{t,i}^{\text{HSMM}} \triangleq P(i, \boldsymbol{\xi}_1 \dots \boldsymbol{\xi}_t | \theta_h)$. Given the model parameters θ_h and the partial observation sequence $\boldsymbol{\xi}_1 \dots \boldsymbol{\xi}_t$, the probability of a datapoint $\boldsymbol{\xi}_t$ to be in state i at time t is recursively computed in the explicit-duration HSMM as [3]

$$\alpha_{t,i}^{\text{HSMM}} = \sum_{j=1}^{K} \sum_{s=1}^{\min(s^{\max}, t-1)} \alpha_{t-s,j}^{\text{HSMM}}\, a_{j,i}\, \mathcal{N}(s|\mu_i^S, \Sigma_i^S) \cdot \prod_{c=t-s+1}^{t} \mathcal{N}(\boldsymbol{\xi}_c | \tilde{\boldsymbol{\mu}}_i, \tilde{\boldsymbol{\Sigma}}_i). \quad (10)$$

The forward variable is initialized with the current state of the task $\boldsymbol{\xi}_{t_o}$ using $\alpha_{t_o,i}^{\text{HSMM}} = \frac{\pi_i \mathcal{N}(\boldsymbol{\xi}_{t_o} | \tilde{\boldsymbol{\mu}}_i, \tilde{\boldsymbol{\Sigma}}_i)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{\xi}_{t_o} | \tilde{\boldsymbol{\mu}}_k, \tilde{\boldsymbol{\Sigma}}_k)}$, and is subsequently used to plan the movement sequence for the next T steps with $t = (t_o + 1) \dots T$. Note that only the transition matrix and the duration model are used to plan the future evolution of the initial/current state $\boldsymbol{\xi}_{t_o}$ (the influence of the spatial data given by the last term is omitted as it has not been observed), i.e. $\mathcal{N}(\boldsymbol{\xi}_t | \tilde{\boldsymbol{\mu}}_i, \tilde{\boldsymbol{\Sigma}}_i) = 1$ for $t = (t_o + 1) \dots T$. This is used to retrieve a stepwise reference trajectory $\mathcal{N}(\hat{\boldsymbol{\mu}}_t, \hat{\boldsymbol{\Sigma}}_t)$ from the state sequence z_t computed from the forward variable, with

$$z_t = \arg\max_i \ \alpha_{t,i}^{\text{HSMM}}, \quad \hat{\boldsymbol{\mu}}_t = \tilde{\boldsymbol{\mu}}_{z_t}^{\mathcal{O}}, \quad \hat{\boldsymbol{\Sigma}}_t = \tilde{\boldsymbol{\Sigma}}_{z_t}^{\mathcal{O}}. \quad (11)$$

E. Linear Quadratic Tracker/Regulator

The desired pose in the shared control mode or the stepwise desired sequence of poses in the autonomous control mode is respectively tracked with an infinite or a finite horizon discrete-time linear quadratic regulator [33]. The cost function minimized during tracking is expressed as

$$c_t(\bar{\boldsymbol{\xi}}_t^{\mathcal{I}}, \boldsymbol{u}_t) = \sum_{t=t_0}^{T} (\bar{\boldsymbol{\xi}}_t^{\mathcal{I}} - \bar{\boldsymbol{\mu}}_t)^\top \boldsymbol{Q}_t (\bar{\boldsymbol{\xi}}_t^{\mathcal{I}} - \bar{\boldsymbol{\mu}}_t) + \boldsymbol{u}_t^\top \boldsymbol{R}_t \boldsymbol{u}_t,$$
$$\text{s.t.} \quad \bar{\boldsymbol{\xi}}_{t+1}^{\mathcal{I}} = \boldsymbol{A}_d \bar{\boldsymbol{\xi}}_t^{\mathcal{I}} + \boldsymbol{B}_d \boldsymbol{u}_t,$$

starting from the initial value $\bar{\boldsymbol{\xi}}_{t_0}^{\mathcal{I}} = \begin{bmatrix} \boldsymbol{\xi}_{t_0}^{\mathcal{I}\top} & \mathbf{0}^\top \end{bmatrix}^\top$, with $\bar{\boldsymbol{\xi}}_t^{\mathcal{I}} = \begin{bmatrix} \boldsymbol{\xi}_t^{\mathcal{I}\top} & \boldsymbol{\xi}_{t+1}^{\mathcal{I}\top} \end{bmatrix}^\top$, $\bar{\boldsymbol{\mu}}_t = \begin{bmatrix} \hat{\boldsymbol{\mu}}_t^\top & \mathbf{0}^\top \end{bmatrix}^\top$, $\boldsymbol{A}_d = \begin{bmatrix} \boldsymbol{I} & \Delta t \boldsymbol{I} \\ \mathbf{0} & \boldsymbol{I} \end{bmatrix}$ and $B_d = \begin{bmatrix} \frac{1}{2}\Delta t^2 \boldsymbol{I} \\ \Delta t \boldsymbol{I} \end{bmatrix}$. Note that the discrete form of a double integrator system is used as a simplified analogue of the robot arm dynamics. Setting $\boldsymbol{Q}_t = \begin{bmatrix} \hat{\boldsymbol{\Sigma}}_t^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \succeq 0$, $\boldsymbol{R}_t \succ 0$, the control input $\boldsymbol{u}_t^* \in \mathbb{R}^7$ that minimizes the cost function is obtained by solving the dynamic Riccati equation backwards in time. For the infinite horizon case with $T \to \infty$ and the desired pose $\hat{\boldsymbol{\mu}}_t = \hat{\boldsymbol{\mu}}_{t_0}$, the control law is obtained by eigendecomposition of the discrete algebraic Riccati equation. The resulting path $\boldsymbol{\xi}_t^{*\mathcal{I}}$ smoothly tracks the desired pose/trajectory $\hat{\boldsymbol{\mu}}_t$ and the computed gains stabilize $\boldsymbol{\xi}_t^{\mathcal{I}}$ along $\boldsymbol{\xi}_t^{*\mathcal{I}}$ in accordance with the precision, changing during the task. The two arms are clutched during teleoperation, and the remote arm is teleoperated under unilateral control mode, i.e., no force is fed back to the teleoperator. Using a haptic interface to feedback interaction forces on the teleoperation site is subject to future work.

III. Comparison with Virtual Fixtures

Virtual fixtures are used to constrain the movement of the remote arm to follow a desired trajectory [8], [9]. The end-effector of the teleoperator arm is virtually coupled to the desired trajectory by a spring-damper system. Like a cart being pulled on a rail, the teleoperator arm movement induces the motion of the remote arm along the trajectory. The desired remote arm pose along the trajectory $\hat{\boldsymbol{\mu}}_{s_{vm}}$ is specified by the phase variable s_{vm} with $s_{vm} = 0$ at the beginning of the trajectory, $s_{vm} = 1$ at the end of the trajectory, and $\dot{\hat{\boldsymbol{\mu}}}_{s_{vm}} = \boldsymbol{J}_{s_{vm}} \dot{s}_{vm}$ where $\boldsymbol{J}_{s_{vm}} \in \mathbb{R}^7$ is the mapping Jacobian. The teleoperator arm movement $\boldsymbol{\xi}_t^{\mathcal{I}}$ induces the dynamics on the the phase variable [22] with

$$\dot{s}_{vm} = \left(\boldsymbol{J}_{s_{vm}}^\top \boldsymbol{B}_\sigma \boldsymbol{J}_{s_{vm}}\right)^{-1} \boldsymbol{J}_{s_{vm}}^\top \Big(\boldsymbol{K}_\sigma (\boldsymbol{\xi}_t^{\mathcal{I}} - \hat{\boldsymbol{\mu}}_{s_{vm}}) + \boldsymbol{B}_\sigma \dot{\boldsymbol{\xi}}_t^{\mathcal{I}} \Big), \quad (12)$$

where $\boldsymbol{K}_\sigma$ and $\boldsymbol{B}_\sigma$ define the stiffness and damping of the virtual fixture.

A task-parameterized HSMM can be used as a virtual fixture by augmenting the teleoperator data with the phase variable s_{vm} during the demonstration step. In the teleoperation phase, the desired remote arm pose is retrieved by Gaussian conditioning on the phase variable (see Eq. (5)) with $\mathcal{P}(\hat{\boldsymbol{\mu}}_{s_{vm}} | s_{vm}) \approx \mathcal{N}(\tilde{\boldsymbol{\mu}}_t^{\mathcal{O}}, \tilde{\boldsymbol{\Sigma}}_t^{\mathcal{O}})$ while the Jacobian $\boldsymbol{J}_{s_{vm}}$ is obtained by evaluating the analytical derivative of Eq. (5) with respect to s_{vm}. Note that the input component here is s_{vm} and the output component $\mathcal{N}(\tilde{\boldsymbol{\mu}}_t^{\mathcal{O}}, \tilde{\boldsymbol{\Sigma}}_t^{\mathcal{O}})$ gives the desired pose $\hat{\boldsymbol{\mu}}_{s_{vm}}$ and its uncertainty, along with the Jacobian $J_{s_{vm}}$ that governs the evolution of the phase variable in Eq. (12) to guide the arm along the trajectory.

In virtual fixture control, the teleoperator arm movement governs the evolution of the phase variable and Gaussian conditioning on the phase variable gives the desired pose of the remote arm; whereas in shared control, Gaussian conditioning on the teleoperator arm pose gives the desired pose of the remote arm. In our implementation of virtual guides, more datapoints were required than in shared control to ensure smooth evolution of the phase variable during teleoperation. We used the logistic function for the phase variable (instead of the linear ramp function) to slow down the cart at the beginning and at the end of the trajectory. The transformation of the phase variable is important to limit injection of arbitrarily high velocities in Eq. (12).

IV. Experiments, Results and Discussions

In this section, we evaluate the semi-autonomous teleoperation approach for reaching a movable target and opening a valve with the Baxter robot. Our experimental protocol remains the same as in Sec. II-A. All the demonstrations

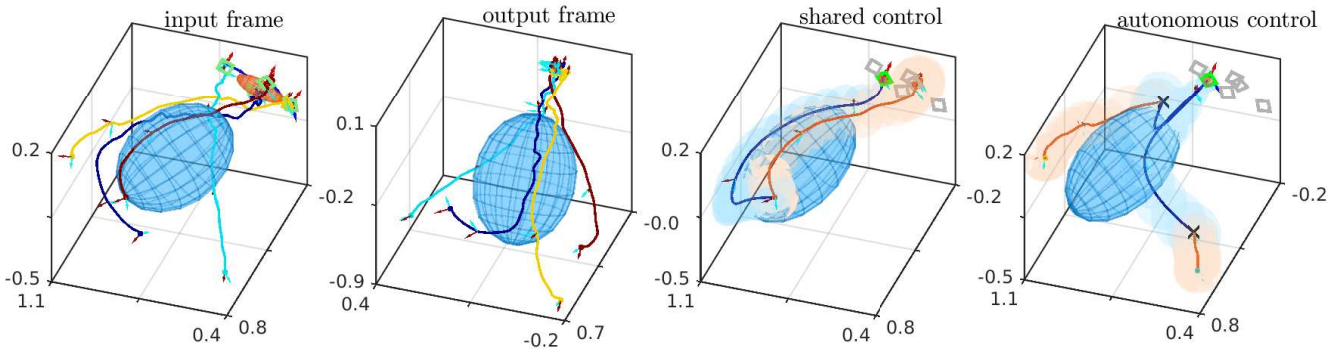

Fig. 3: Reaching a movable target (in green squares) for screw driving: *(left)* demonstrations and model shown in the input reference frame; *(center-left)* demonstrations and model in the output reference frame; *(center-right)* the teleoperator demonstration (in red) is corrected under shared control (in blue) to reach a new target location shown in green; *(right)* the teleoperator switches from direct control to autonomous control mode (marked with a cross) after which the movement is autonomously generated to the new target location.

are collected with a controller compensating for the effect of gravity by a human operator who is familiar with the robot, but not an expert in teleoperation. For learning, the number of Gaussians is empirically selected for each task based on a crude division of the task into phases such as reaching, grasping, etc. Alternatively, a Bayesian information criterion, or a non-parametric approach based on Dirichlet processes could be used for model selection [34], [27]. Performance setting in all our experiments is as follows: the mixture components are initialized as left-right HSMM using k-means clustering. We evaluate the performance of our approach using three different criteria: 1) task performance error, 2) environmental differences, and 3) execution time.

A. Manipulation Tasks

1) Reaching a Movable Object: The objective of this task is to reach a target point with a screwdriver while adapting the movement for different target configurations. We describe the task with a single frame $\{\boldsymbol{A}_1, \boldsymbol{b}_1\}$ attached to the target and collect 6 kinesthetic demonstrations (4 for training and 2 for testing) with the initial pose of the target rotated/translated in the successive demonstrations. Results of the joint distribution with 2 mixture components for different target poses are shown in Fig. 3. Demonstrations for the input reference frame represent the movement of the end-effector of the teleoperator's arm to different target poses, while the output reference frame maps the demonstrations to a pose as observed from the target perspective.

2) Opening a Valve: The goal of this task is to bring the valve in an open position from different initial configurations of the valve [4]. The task is described by two reference frames, one with the observed initial configuration of valve $\{\boldsymbol{A}_1, \boldsymbol{b}_1\}$ and the other with the desired end configuration of the valve $\{\boldsymbol{A}_2, \boldsymbol{b}_2\}$. The changing configuration of the valve is tracked using an augmented reality (AR) tag with a Kinect 2.0. We record 8 kinesthetic demonstrations (6 for training and 2 for testing) with the initial configuration of the valve corresponding to $\{180, 135, 90, 45, 157.5, 112.5, 67.5, 22.5\}$ degrees with the horizontal in the successive demonstrations. Results of the learned model with 2 reference frames and 7 mixture components are shown in Fig. 4. The input components of both reference frames represent the demonstrations identically in global coordinates. The output components of the reference frames depict high variability in reaching the valve and coming back to the home position, whereas there is no variation in grasping/turning and stopping the valve in their respective coordinate systems. This allows the robot arm to reach the valve from different configurations, grasp the valve and turn it to the desired open position.

B. Performance Evaluation

1) Task Performance Error: Our objective is to assist the teleoperator to perform remote manipulation tasks in a repeatable and precise manner while reducing the workload of the teleoperator. Results of the shared and autonomous modes of assistance for target reaching task are shown in Fig. 3. In shared control, the model corrects the movement of the teleoperator in accordance with the output component of the model that adapts locally to the target. If the variance of the resulting conditional distribution is low, the correction is stronger and vice versa. While demonstrating autonomous control, the teleoperator tests the system by randomly switching between control modes during the task. Fig. 3 (*right*) shows how the movement of the robot converges to the target from different switching instants, while being repeatable and more precise than the direct and the shared control results.

2) Robustness to Different Environments: Performance of the teleoperator is typically affected by the environmental differences between the teleoperator and the remote sites. Such differences exist as streaming full OctoMaps over satellite communication for updating the remote environment on the teleoperator site are only possible at a very low frequency. In Fig. 4, we compare different assistance approaches to handle these discrepancies by setting different configurations of the valve on the teleoperator and the remote end. We can see that the task-parameterized model successfully adapts to the external situation on the teleoperator and the remote site, thereby, mitigating the difference of situations in the

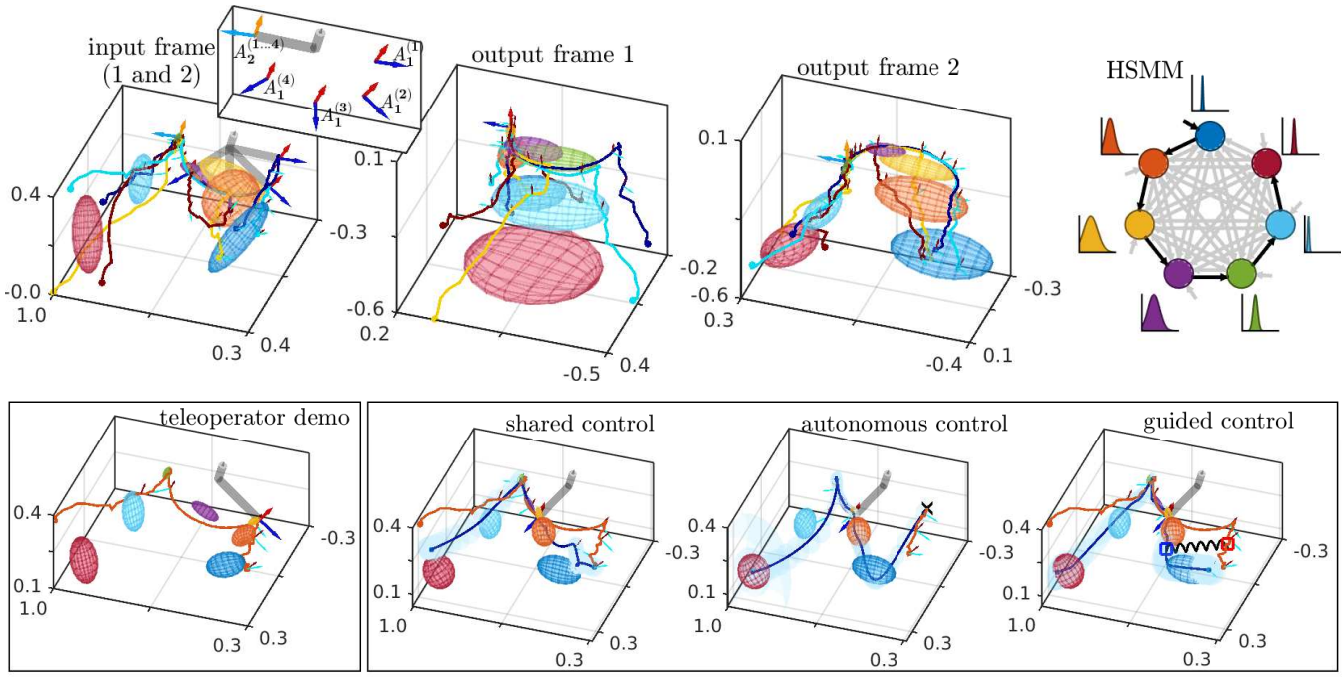

Fig. 4: Open valve (in gray) from different initial configurations $\boldsymbol{A}_1^{(i)}$ to final configuration $\boldsymbol{A}_2$: *(top)* learned model in the input and output reference frames, and left-right HSMM with state transition and state duration model ($s^{\max} = 100$); *(bottom)* the teleoperator performs the task (in red) with respect to the perceived valve configuration on the left, where the different control modes assist the remote arm (in blue) to perform the task with actual valve configuration. The spring in bottom right is used to show the virtual fixture between the teleoperator pose and the desired pose along the trajectory.

two sites. The teleoperator performs the movement according to the perceived valve configuration or switches to the autonomous mode while performing the task, and the generated movement is adapted locally with shared, autonomous or virtual fixture control.

Table I summarizes the results of different control modes to mitigate imprecise teleoperator movements with our model (see [28] for comparison with other models). For each target or valve configuration in the training or testing set, all the demonstrations in the training and testing sets are treated in a given control mode and compared with the human demonstration for that particular target. Mean-squared end-point errors for target tracking and mean-squared trajectory errors for valve opening tasks are averaged over all demonstrations and for all target or valve configurations. The results show that the autonomous control gives the most repeatable and precise assistance among different teleoperation modes, while modeling the data in latent space provides a comparable performance with much less parameters. Moreover, we observe high performance errors of virtual fixture control on valve opening task as change of movement directions in the teleoperator demonstration tends to induce remote arm movement in the reverse direction along the trajectory, resulting in an unsuccessful trial.

3) Execution Time: In order to evaluate the effect of teleoperation mode on the task execution time, the human operator performs the task 5 times for each teleoperation mode from different initial conditions. At the end of all trials, the operator reveals the preferred mode of assistance for each task. Results in Table II suggest performance improvement in task execution time using the learned model as compared to the direct teleoperation mode. In our future work, we plan to test the model with large communication delays over satellite communication in challenging underwater environments. Moreover, we would like to systematically evaluate the performance of the teleoperator in executing the task under delayed feedback from the remote site.

V. Conclusion

In this paper, we have exploited a task-parameterized HSMM to assist the teleoperator in performing remote manipulation tasks. We have presented probabilistic formulations of the model to provide assistance by correcting the robot motion with shared control, synthesizing the motion with autonomous control, or following a desired trajectory with virtual fixture control. Compared to synchronous direct teleoperation, the proposed semi-autonomous teleoperation framework locally adapts the model to handle environmental differences and communication delays. Teleoperation experiments conducted with the Baxter robot for target reaching and valve opening tasks showed improvement in mitigating imprecise teleoperator demonstrations and reducing the task execution time.

References

[1] J. Gancet *et al.*, "DexROV: Dexterous Undersea Inspection and Maintenance in Presence of Communication Latencies," in *IFAC workshop on NGCUV*, 2015, pp. 218–223.

TABLE I: Performance comparison of the teleoperation modes with direct control (DC), shared control (SC), autonomous control (AC), and virtual fixture control (VFC). N_p is the number of parameters of the model, and the errors represent average mean squared errors between the demonstrations and the model predictions (in centimeters).

N_p	SC		AC		VFC	
	Train	**Test**	**Train**	**Test**	**Train**	**Test**
valve opening ($K = 7, D = 14$, **DC Train =** 1.412 ± 1.20, **DC Test =** 1.261 ± 1.13)						
1736	0.717 ± 0.67	0.721 ± 0.68	0.737 ± 0.62	0.464 ± 0.33	2.836 ± 2.07	2.011 ± 0.83
target snapping ($K = 2, D = 14$, **DC Train =** 1.954 ± 2.04, **DC Test =** 1.959 ± 1.81)						
248	0.23 ± 0.25	0.31 ± 0.26	0.109 ± 0.08	0.178 ± 0.09	0.183 ± 0.12	0.311 ± 0.27

TABLE II: Average execution time of performing a manipulation task in seconds under different teleoperation modes.

Task	DC	SC	AC	VFC	Mode
valve opening	18.2	13.6	**12.3**	13.1	**AC**
target snapping	7.2	**4.3**	5.1	5.6	**SC**

[2] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–285, 1989.

[3] S.-Z. Yu, "Hidden semi-Markov models," *Artificial Intelligence*, vol. 174, pp. 215–243, 2010.

[4] A. K. Tanwani and S. Calinon, "Learning robot manipulation tasks with task-parameterized semi-tied hidden semi-Markov model," *IEEE Robotics and Automation Letters (RA-L)*, vol. 1, no. 1, pp. 235–242, January 2016.

[5] G. Niemeyer, C. Preusche, and G. Hirzinger, *Telerobotics*. Springer Berlin Heidelberg, 2008, pp. 741–757.

[6] P. F. Hokayem and M. W. Spong, "Bilateral teleoperation: An historical survey," *Automatica*, vol. 42, no. 12, pp. 2035–2057, 2006.

[7] T. Sheridan, "Teleoperation, telerobotics and telepresence: A progress report," *Control Engineering Practice*, vol. 3, no. 2, pp. 205–214, 1995.

[8] L. B. Rosenberg, "Virtual fixtures: Perceptual tools for telerobotic manipulation," in *IEEE Virtual Reality Annual International Symposium (VRAIS)*, 1993, pp. 76–82.

[9] J. J. Abbott, P. Marayong, and A. M. Okamura, "Haptic virtual fixtures for robot-assisted manipulation," in *Proc. Intl Symp. on Robotics Research (ISRR)*, 2007, pp. 49–64.

[10] T. B. Sheridan, *Telerobotics, Automation, and Human Supervisory Control*. Cambridge, MA, USA: MIT Press, 1992.

[11] D. Yoerger and J. J. Slotine, "Supervisory control architecture for underwater teleoperation," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, vol. 4, Mar 1987, pp. 2068–2073.

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

[13] A. G. Billard, S. Calinon, and R. Dillmann, "Learning from humans," in *Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Secaucus, NJ, USA: Springer, 2016, ch. 74, pp. 1995–2014, 2nd Edition.

[14] L. Rozo, S. Calinon, D. G. Caldwell, P. Jimenez, and C. Torras, "Learning physical collaborative robot behaviors from human demonstrations," *IEEE Trans. on Robotics*, vol. 32, no. 3, pp. 513–527, 2016.

[15] W. Yu, R. Alqasemi, R. Dubey, and N. Pernalete, "Telemanipulation assistance based on motion intention recognition," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2005, pp. 1121–1126.

[16] A. D. Dragan and S. S. Srinivasa, "A policy-blending formalism for shared control," *I. J. Robotic Res.*, vol. 32, no. 7, pp. 790–805, 2013.

[17] K. Hauser, "Recognition, prediction, and planning for assisted teleoperation of freeform tasks," *Autonomous Robots*, vol. 35, no. 4, pp. 241–254, 2013.

[18] D. Aarno and D. Kragic, "Motion intention recognition in robot assisted applications," *Robot. Auton. Syst.*, vol. 56, no. 8, pp. 692–705, Aug. 2008.

[19] A. Pentland and A. Liu, "Modeling and prediction of human behavior," *Neural Comput.*, vol. 11, no. 1, pp. 229–242, Jan. 1999.

[20] M. Li and A. M. Okamura, "Recognition of operator motions for real-time assistance using virtual fixtures," in *Proc. of 11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2003, pp. 125–131.

[21] J. T. Nolin, P. M. Stemniski, and A. M. Okamura, "Activation cues and force scaling methods for virtual fixtures," in *Proc. of 11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2003, pp. 404–409.

[22] G. Raiola, X. Lamy, and F. Stulp, "Co-manipulation with multiple probabilistic virtual guides," in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, 2015, pp. 7–13.

[23] J. Medina, M. Lawitzky, A. Mortl, D. Lee, and S. Hirche, "An experience-driven robotic assistant acquiring human knowledge to improve haptic cooperation," in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, 2011, pp. 2416–2422.

[24] Z. Wang, K. Mülling, M. Deisenroth, H. Ben Amor, D. Vogt, B. Schölkopf, and J. Peters, "Probabilistic movement modeling for intention inference in human-robot interaction," *International Journal of Robotics Research*, vol. 32, no. 7, pp. 841–858, 2013.

[25] G. Maeda, G. Neumann, M. Ewerton, R. Lioutikov, and J. Peters, "A probabilistic framework for semi-autonomous robots based on interaction primitives with phase estimation," in *International Symposium of Robotics Research*, 2015.

[26] A. K. Tanwani and S. Calinon, "Online inference in Bayesian non-parametric mixture models under small variance asymptotics," in *NIPS workshop on Advances in Approximate Bayesian Inference*, 2016, pp. 1–5.

[27] A. K. Tanwani and S. Calinon, "Small variance asymptotics for non-parametric online robot learning," *Int. J. of Rob. Research IJRR*, vol. abs/1610.02468, 2016, (conditionally accepted). [Online]. Available: http://arxiv.org/abs/1610.02468

[28] A. K. Tanwani, "Generative Models for Learning Robot Manipulation Skills from Humans," Ph.D. dissertation, Ecole Polytechnique Fédérale de Lausanne (EPFL), 2017, (in press).

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

[30] Z. Ghahramani and M. I. Jordan, "Supervised learning from incomplete data via an EM approach," in *Advances in Neural Information Processing Systems*, vol. 6, 1994, pp. 120–127.

[31] F. Abi-Farraj, T. Osa, N. Pedemonte, J. Peters, G. Neumann, and P. Giordano Robuffo, "A Learning-based Shared Control Architecture for Interactive Task Execution," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Singapore, 2017, pp. 329–335.

[32] E. Pignat and S. Calinon, "Learning adaptive dressing assistance from human demonstration," *Robotics and Autonomous Systems*, vol. 93, pp. 61–75, July 2017.

[33] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2011.

[34] I. Havoutis, A. K. Tanwani, and S. Calinon, "Online incremental learning of manipulation tasks for semi-autonomous teleoperation," in *IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS), Workshop on Closed-loop Grasping and Manipulation: Challenges and Progress*, Daejeon, Kora, October 2016.