Online correction of task registration and robot models from user input

Thibault Poignonec, Florent Nageotte, Nabil Zemiti, Bernard Bayle

Online correction of task registration and robot models from user input

THIBAULT POIGNONEC*, ICube laboratory, University of Strasbourg-CNRS-INSERM, France
FLORENT NAGEOTTE, ICube laboratory, University of Strasbourg-CNRS-INSERM, France
NABIL ZEMITI, LIRMM laboratory, CNRS, University of Montpellier, France
BERNARD BAYLE, ICube laboratory, University of Strasbourg-CNRS-INSERM, France

In application domains such as surgical robotics, fully autonomous control remains a long-term ambition and the systems are mostly teleoperated. In this article, the presence of an operator in-the-loop is exploited to perform the online registration of an initially inaccurate haptic guidance and the calibration of robot kinematic models using operator's intention instead of relying on exteroceptive sensors. This is used to improve online haptic guidance in the context of shared control, or to progress towards automatic task completion after an initial learning phase. The method presented in this paper is based on an optimization in the task space to minimize the errors between the executed and desired trajectories, both estimated from models. This approach is particularly relevant when the execution of a planned task would suffer from errors that exteroceptive measurements could not fully correct, because of sensor inaccuracy or unavailability. A user study realized for a drawing task is detailed to illustrate that initially inaccurate task registration and robot models can be corrected from user inputs only. The results show that the proposed algorithm can learn the correct models, which in turns significantly improves the quality of the haptic guidance and decreases path deviations during the teleoperated task.

CCS Concepts: • **Computer systems organization → Robotic control**; • **Computing methodologies → Online learning settings**.

Additional Key Words and Phrases: Online learning, Registration, Haptic guidance

1 INTRODUCTION

Many robotic tasks involve the planning of a path and the subsequent execution of the associated trajectory, which involves time parameterization. In such a procedure, a registration step is necessary to transfer the planned trajectory to the robot's operational space. Significant registration errors, which can happen for instance when the environment is not structured or when there are few landmarks in the scene, result in inaccurate task execution. Even if the planning and registration are correct, robot modeling errors can also lead to inaccurate task execution. Such errors can occur when grasping tools during the task, for instance. The correction of both task registration and robot modeling is of particular importance for the automatic execution of robotic tasks. One could think that the accuracy of interactive tasks might not be as much affected since the human operator involved in the task could at least partially correct such errors. However, appropriate registration, as well as accurate robot models, are also crucial in several interactive scenarios, for instance when using visual guidance

Authors' addresses: Thibault Poignonec, tpoignonec@unistra.fr, ICube laboratory, University of Strasbourg-CNRS-INSERM, Strasbourg, France; Florent Nageotte, ICube laboratory, University of Strasbourg-CNRS-INSERM, Strasbourg, France, nageotte@unistra.fr; Nabil Zemiti, LIRMM laboratory, CNRS, University of Montpellier, Montpellier, France, nabil.zemiti@lirmm.fr; Bernard Bayle, ICube laboratory, University of Strasbourg-CNRS-INSERM, Strasbourg, France, bernard.bayle@unistra.fr.

ACM 2573-9522/2024/10-ART
https://doi.org/10.1145/3702246

with augmented reality [4, 42] or force feedback virtual fixtures (VF) [5, 7, 22, 39] . Indeed, while such methods are expected to guide the user in order to execute the task or reduce cognitive load [11], inaccurate guidance can notably disturb the user [30].

The previous problems are particularly relevant in the medical context. Interventions planning very often uses pre-operative images that are no longer available intra-operatively or different from the ones available [49, Chapter 3]. The registration step is, therefore, challenging. In medical robotics, mechanical parts and transmissions are often not rigid [35, 38], and surgical instruments grasping configuration is generally not well known [25]. In situ model estimation is then necessary to improve automatic task achievement or surgeon guidance. Exteroceptive measurements could be used to solve these problems, typically by exploiting visual features [21]. Solutions using computer vision [12, 24, 29, 43] or force sensing [44, 51, 53] have already been proposed to update the models online, but such measurements can be very difficult to obtain in unstructured environments, and could even become unavailable, at least intermittently.

To cope with this problem, the presence of the operator to perform the task remains necessary. As a result, most surgical robotic systems are teleoperated. This could be made profitable to solve registration and modeling issues. The present article aims to exploit the user actions to improve robotic assistance by the online correction of such errors resulting in deviations from preoperatively planned trajectory. Some surgical tasks are particularly complex and dependent on the in situ conditions of the realization and thus cannot easily be described by a planned trajectory. Therefore they do not fall into the scope of the proposed method. However, many medical procedures, typically the most codified ones, include or sometimes require planned trajectories. To cite only a few examples, we can mention suturing tasks [17, 25, 31], path following in Transcranial Magnetic Stimulation [55], or optimal area covering for blood-brain barrier opening [48]. While such medical applications inspired this research, the present article does not have the ambition to be directly applied to a medical procedure, which would require specific precautions, as discussed later. In addition, the possibility of using human input to correct task or robot modeling errors could benefit the robotics community more generally, as exemplified by past publications such as [2, 19].

Using the presence of the user "in the loop" to correct modeling errors has already deserved some attention in the literature, with different points of view. On the one hand, some authors previously proposed to observe the operator actions in order to select, among a library of predefined primitives, the trajectory that best explains these actions. This approach was used to generate adaptive VF [1] or to complete the rest of the task automatically [54]. In some works [9, 13, 23], the authors used human input to deform a previously planned task during its automatic execution, but the operator is then relegated to a supervisory role. Methods were proposed to adapt a task defined by a goal [15] or by a choice among predefined trajectories [36], but these methods are not directly applicable to adapt the trajectories themselves in real-time. On the other hand, the correction of trajectory execution errors from physical human-robot interaction has also received notable attention in recent years. However, in the proposed solutions, the learning is usually done iteratively and, therefore, not always adapted to situations where the models can change during the task execution. Task adaption was achieved by the iterative minimization of the interaction forces applied to the robot by the operator correcting the behavior [52] or by modifying a path to match demonstrated corrections [40]. Iterative implementations from operator actions were also proposed for learning from demonstration frameworks as in [16]. Online solutions were proposed to deform a trajectory during collaborative manipulation using interactions of an operator with the robot, both to deform a pre-planned trajectory locally [20, 27, 41] or globally, through parameters updates [19]. The method proposed in [20] was extended to refine the task model from deformed trajectories [2, 6, 18]. This method focuses on learning parameters-linear reward functions that can then be used to replan the trajectory in real-time. However, they are not designed to refine the potentially inaccurate task registration. Furthermore, the point-to-point procedure assumes that the time parameterization of the trajectory is correct, which is generally not true when the robot is teleoperated by an operator who imposes the pace.

A learning method to refine task parameters used for the registration of the path in the robot frame was proposed in one of our previous works [34], based on observed robot trajectories during teleoperation. It allows to correct the task model online and to provide adaptive haptic guidance. However, this preliminary study was not presented in a general framework. In particular, like all the methods mentioned so far, it only considers task model corrections under the assumption that the robot kinematic model is perfectly known or that adequate measurements are available. In practice, the robot model might also be partially inaccurate, which invalidates the planning of the task expressed in the joint space and leads to execution errors. Although some methods were proposed to adapt the robot model online using teleoperation execution, the state of the robot is usually fully measured [8]. To the best of our knowledge, simultaneous online learning of task and robot kinematic parametric models has received little attention to date. The published method closest to what is intended here extracts both the correct task and robot kinematic models from demonstrations [32], but the learning is not performed online when the operator interacts with the robot to perform the task.

The contribution of the present paper is to propose an online learning framework to exploit the information provided by an operator in-the-loop, in order to refine task registration and robot model parameters. It is essential to clarify that this contribution is a methodological work inspired by a particular, but relevant problem : the guidance of gestures in teleoperated robotic surgery. In this field, as in others where safety issues are at stake, a fully autonomous system control remains a long-term ambition. As things stand, guiding the operator would be of considerable interest compared with conventionally available systems. In the present paper, we considered it necessary to carefully study solutions to the methodological problem. We then propose a method inspired by medical needs and constraints, but in a controlled environment, with a task of lesser difficulty than a complex medical task. This enables a more reliable analysis of the method itself.

This article is organized as follows. Section 2 presents the problem and the assumptions necessary to develop the approach, which is detailed in Section 3, in a generic way. In order to validate the approach and the efficiency of the parameters learning, a path-following task with online adapted haptic guidance is performed, with a general-purpose robot. This experimental setup, described in Section 4, is particularly well suited to provide objective task performance assessment. The last sections respectively report the obtained experimental results, in Section 5 and 6, before a detailed discussion concludes the article, in Section 7.

2 PROBLEM STATEMENT

Before detailing the proposed approach in Section 3, we introduce the considered telerobotics scenario along with the different assumptions and models that will be used throughout this section.

2.1 Considered setup and modeling

We consider a robot operated remotely in order to perform a trajectory-following task. Let $x_{l,k}$ be the Cartesian pose of the leader robot in its frame of reference $\mathcal{F}_l$ and $x_{f,k}$ be the follower robot Cartesian pose in a world frame $\mathcal{F}_w$ attached to its base. The notation $x_{l,k}$ denotes discrete time such that $x_{l,k} = x_l(t_k)$, with $t_k = kT_s$ and T_s the sampling period. The leader robot pose $x_{l,k}$ is mapped to the follower operational workspace through a linear map $\mathcal{M}(.)$ to generate the follower robot Cartesian reference $x^r_{f,k}$. From this reference, a position controller computes the follower robot joint position reference that is assumed to be ideally tracked by a low-level controller. The forward kinematic model of the follower robot is written as $x_{f,k} = \mathcal{K}(\theta_{r,k}, q_{f,k})$, where $\theta_{r,k}$ is a vector of robot model parameters and $q_{f,k}$ denotes the joint positions. The position controller operates based on an estimate of the parameters $\hat{\theta}_{r,k}$ resulting in an estimated follower pose

$$\hat{x}_{f,k} = \mathcal{K}(\hat{\theta}_{r,k}, q_{f,k}) \tag{1}$$

This yields the kinematic relation described on the block diagram given in Figure 1. The estimated robot parameters $\hat{\theta}_{r,k}$ used by the controller can either be fixed (e.g., computed beforehand from a calibration process or given by the robot data sheet) or updated during the execution of the task. This whole process can be seen as a possibly non-linear mapping from $x_{l,k}$ to the follower pose

$$x_{f,k} = \Psi\left(x_{l,k}\right) \tag{2}$$

where

$$\begin{aligned}\Psi(x_{l,k}) &= \mathcal{K}\left(\theta_{r,k}, \mathcal{K}^{-1}(\hat{\theta}_{r,k}, x^r_{f,k})\right) \\ &= \mathcal{K}\left(\theta_{r,k}, \mathcal{K}^{-1}(\hat{\theta}_{r,k}, \mathcal{M}(x_{l,k}))\right)\end{aligned} \tag{3}$$

The term $\mathcal{K}^{-1}$ denotes the reciprocal function of the forward kinematic model. It is used here mainly for helping the reader to understand the general concept. The use of the reciprocal is implicit in later developments. In practice, this inverse function used by the robot controller can be solved numerically, for instance with a pseudo-inverse or through an appropriate optimization method.

Note that the choice of directly modeling the kinematic chain of the robot and not using a differential model affects the scope of the inaccuracies that can be captured through parameterization with $\theta_{r,k}$. The experimental validation presented in this work treats the common case of an inaccurate tool registration w.r.t. the end effector of a rigid manipulator robot. Although this is the most prominent form of error that can be modeled by equation (1), other errors that can be readily considered including joint position offsets, inaccurate transmission ratio estimates, or inaccurate link length. In some cases, even position inaccuracies caused by non-rigid elements can lead to a parameterized model of the kinematic chain. For instance, in the context of medical robotics, the mechanical backlash resulting from the cable actuation of flexible endoscopes can be approximated using a parametric (discontinuous) kinematics model [50], which leads to a mapping between leader and follower similar to the one in Figure 1. In a previous work by the authors [33], a method was proposed to learn online the parameters of this type of model.

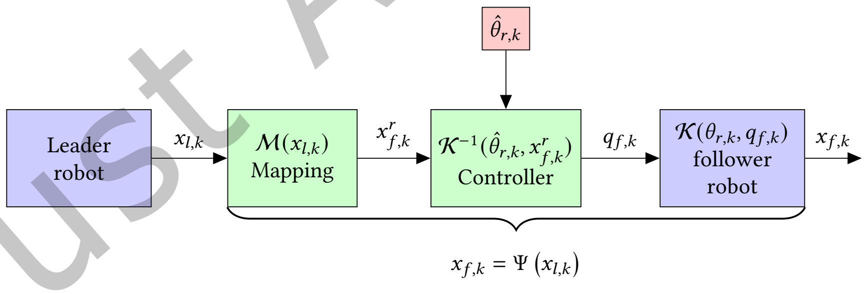
$\hat{\theta}_{r,k}$
Leader robot
$x_{l,k}$
$\mathcal{M}(x_{l,k})$ Mapping
$x^r_{f,k}$
$\mathcal{K}^{-1}(\hat{\theta}_{r,k}, x^r_{f,k})$ Controller
$q_{f,k}$
$\mathcal{K}(\theta_{r,k}, q_{f,k})$ follower robot
$x_{f,k}$
$x_{f,k} = \Psi\left(x_{l,k}\right)$

Fig. 1. Block diagram of the position control of the follower robot. The whole process can be seen as a non-linear mapping $x_{f,k} = \Psi\left(x_{l,k}\right)$ from leader pose $x_{l,k}$ to effective follower robot pose $x_{f,k}$ that depends on both the estimated robot model parameters used by the controller and the real robot model parameters.

2.2 User desired trajectory

The user aims at performing a desired task with the follower robot. To do so the user manipulates the leader robot relying on a real-time visual feedback to determine the desired task. This desired follower robot Cartesian

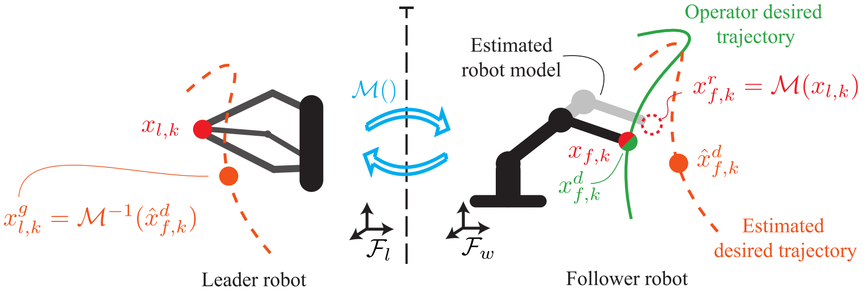

Fig. 2. Teleoperation setup where the operator remotely controls the follower robot to execute a desired trajectory (green). An estimation of this desired trajectory (orange) is used to provide a leader side haptic guidance, though this estimation is incorrect. The resulting guidance error also includes the follower robot positioning errors resulting from erroneous kinematic modeling (gray vs. black).

trajectory expressed in the world frame is denoted $x^d_{f,k}$. It is assumed to be described by a family of parametric curves g such that

$$x^d_{f,k} = g(\theta_{g,k}, t_k) \tag{4}$$

where $\theta_{g,k}$ is a set of task parameters. These parameters should typically encode the time parameterization of the task (e.g., the desired velocity) and its registration from a planning frame to the world frame. Any parameterized expression of the form $g(\theta_{g,k}, t_k)$ can be used, as long as it is differentiable w.r.t. time and parameters.

Using the visual feedback, the user will try to adjust the leader robot such that the follower robot is at the desired pose at all time. Due to perception biases and intrinsic human limitations (e.g., precision, response time, etc.), there is a time-varying execution error $\epsilon_{h,k}$ between the desired and effective follower robot poses such that

$$x_{f,k} = x^d_{f,k} + \epsilon_{h,k} \tag{5}$$

2.3 Haptic guidance

The haptic guidance essentially helps the user to perform the desired trajectory by applying guidance forces or torques to the leader robot. These forces are computed to render a mechanical impedance whose equilibrium position is the guidance reference pose $x^g_{l,k}$, itself evaluated from a model of the operator's desired trajectory. Note that in the context of this work, the parameters of the desired trajectory are not known and have to be estimated. As the desired trajectory is modeled by a family of parametric curves, estimating the desired trajectory amounts to estimating the parameters $\theta_{g,k}$. Therefore, the estimated desired pose in the world frame is

$$\hat{x}^d_{f,k} = g(\hat{\theta}_{g,k}, t_k) \tag{6}$$

where $\hat{\theta}_{g,k}$ is the current estimate of $\theta_{g,k}$. Since the guidance forces are applied to the leader robot, the guidance reference must be mapped back into the leader workspace such that

$$x^g_{l,k} = \mathcal{M}^{-1}\left(\hat{x}^d_{f,k}\right) \tag{7}$$

The complete teleoperation setup is represented in Figure 2 along with the desired trajectory $x^d_{f,k}$ and the guidance reference $x^g_{l,k}$ resulting from Equation (7).

An accurate estimation of the parameters $\theta_{r,k}$ and $\theta_{g,k}$ is critical to the performance of the guidance. It is easy to understand that an inaccurate task parameters estimate would cause the guidance to be inaccurate. The effect of errors on the robot kinematic parameters is probably more difficult to understand, but it is actually similar. Equation (2) reduces to $\Psi(x_{l,k}) = \mathcal{M}(x_{l,k})$ *if and only if* $\hat{\theta}_{r,k} = \theta_{r,k}$ as in this case Equation (2) becomes $x_{f,k} = \mathcal{K}\left(\theta_{r,k}, \mathcal{K}^{-1}(\theta_{r,k}, \mathcal{M}(x_{l,k}))\right)$. The user will be guided towards $x^g_{l,k}$ only under these conditions. An error on the robot model parameters leads to inaccurate guidance, even assuming that the follower robot pose desired by the user $x^d_{f,k}$ is perfectly known, as, from equations (2) and (7) it then comes that:

$$\begin{cases} \hat{\theta}_{r,k} \neq \theta_{r,k} \\ \hat{\theta}_{g,k} = \theta_{g,k} \\ x_{l,k} = x^g_{l,k} \end{cases} \implies \quad x_{f,k} = \Psi\left(\mathcal{M}^{-1}\left(x^d_{f,k}\right)\right) \neq x^d_{f,k}$$

In that case, the guidance is also experienced as inaccurate by the operator who will have to compensate for the follower robot's positioning error. Additionally, if the kinematic model is strongly non-linear or/and the robot modeling errors are large, the trajectory that the user will have to follow at the leader side will be deformed and potentially non-intuitive.

Regardless of the source of the guidance inaccuracies, the user will experience forces pulling towards undesired directions, which is an inconvenience and can even lead to degraded task achievement. Either way, the haptic guidance stops being assistive when the guidance errors increase. It is therefore critical to correct the task and kinematic parameters when either or both are incorrect and to do so online because the optimal parameter values can change during the task execution. In the following, an approach to online parameters learning is presented to cope with task and robot model inaccuracies.

2.4 Learning problem

The learning problem can be stated as two simultaneous parameters estimations that, in their simplest forms, would be written as:

$$\theta^*_r = \arg\min_{\theta_r} \sum_{i=0}^{k} ||x_{f,i} - \mathcal{K}(\theta_r, q_{f,i})||^2 \tag{8}$$

$$\theta^*_g = \arg\min_{\theta_g} \sum_{i=0}^{k} ||x^d_{f,i} - g(\theta_g, t_i)||^2 \tag{9}$$

assuming that the parameters do not change over time for a considered time horizon $[0;\ t_k]$. Note that, however, neither the follower robot's Cartesian pose nor the desired one might be directly measured. Usually, only partial observations are available, if any. Possible sensor information can come from camera image measurements, force sensing, etc., but this information can sometimes be poor or intermittent. It is then advantageous to exploit user inputs because they are dependent on both the real task and robot kinematic parameters.

3 GENERAL METHOD

3.1 User in-the-loop as an information source

The user will manipulate the leader robot in such a way that the follower robot's pose $x_{f,k}$ is as close as possible to the desired position $x^d_{f,k}$. Hence, $x_{f,k}$ can be seen as a noisy observation of $x^d_{f,k}$ as defined in Equation (5). However, if the follower robot Cartesian pose is not measured by an external sensor, the only observation available for learning is the joint position $q_{f,k}$. Then, we propose to use the presence of the operator to formulate an optimization problem that minimizes the discrepancy between the estimation of the executed Cartesian trajectory and the trajectory predicted by the task model.

3.2 General optimization formulation

The general learning problem is then stated as the minimization of a loss function that penalizes the violation of the constraint resulting from Equation (5). The aforementioned constraint is then written as

$$\begin{aligned} &\mathcal{K}(\theta_{r,k}, q_{f,k}) - g(\theta_{g,k}, t_k) = \epsilon_{h,k} \\ &\text{with } \epsilon_{h,k} \sim \mathcal{N}(0, \Sigma_{h,k}) \end{aligned} \tag{10}$$

where $\mathcal{N}(.)$ represents a normal distribution such that $\Sigma_{h,k}$ is the covariance of the (presumably) normally distributed zero-mean error $\epsilon_{h,k}$. Additional observations about the state of the robot and the environment can be incorporated in the form of additional terms in the cost function. Let z_k be a vector containing the observations acquired at time t_k that depend on the parameters and can be predicted as

$$z_k = h(\theta_k) + w_k, \;\; w_k \sim \mathcal{N}(0, R_k) \tag{11}$$

where $h(.)$ is the observation model, $\theta_k = [\theta_{g,k}^T \; \theta_{r,k}^T]^T$ contains the different model parameters, and w_k is the measurement error of covariance R_k. Small parameters variations arising from online changes in the environment, in the kinematics of the robot, or in the preferences of the human operator can be modeled by a Gaussian noise v_k with covariance Q_k such that

$$\theta_{k+1} = \theta_k + v_k, \;\; v_k \sim \mathcal{N}(0, Q_k) \tag{12}$$

The online learning problem can be seen as an optimization performed at time step k considering the measurements $z_{0:k-1}$ and $q_{f,0:k-1}$ acquired at discrete times $t_{0:k-1}$, i.e., the sequence of times $\{t_0; \, t_1, \, \cdots, \, t_{k-1}\}$. The optimization is performed w.r.t. the sequence of parameters $\theta_{0:k}$. If all available observations are considered, it can be written as:

$$\theta^*_{0:k} = \underset{\theta_{0:k}}{\arg\min} \; l_\theta(\theta_0) + \sum_{i=0}^{k-1} l_i(\theta_i, \theta_{i+1}, z_i) \tag{13}$$

with

$$\begin{aligned} l_\theta(\theta_0) &= \|\hat{\theta}_0 - \theta_0\|^2_{P_0^{-1}} \\ l_i(\theta_i, \theta_{i+1}, z_i) &= \left\| \mathcal{K}(\theta_{r,i}, q_{f,i}) - g(\theta_{g,i}, t_i) \right\|^2_{\Sigma_{h,i}^{-1}} + \|\theta_{i+1} - \theta_i\|^2_{Q_i^{-1}} + \|z_i - h(\theta_i)\|^2_{R_i^{-1}} \end{aligned}$$

where the function $l_\theta(\cdot)$ penalizes initial parameters update, with $\hat{\theta}_0$ the initial guess for the parameters value at $k = 0$ and P_0 the covariance of this initial estimate. The term $l_\theta(\cdot)$ allows to include prior knowledge about the initial parameters distribution. Its effect fades over time as more observations are available. The function $l_i(\cdot)$ is a weighted sum of prediction errors computed according to the models defined by equations (10), (11), and (12). The learning framework can be used in conjunction with a vast class of non-linear state/parameters estimation methods such as moving horizon estimation [37], or other non-linear Bayesian estimation techniques (e.g., extended or unscented Kalman filters, particle filters, etc.). This flexible approach allows to use the presence of an operator to augment the information available for learning. Although not required, sensor measurements can be included for learning and, due to the Bayesian formulation, the fusion of observations coming from multiple sensors is streamlined, with the sole requirement that the covariance of the measurements can be estimated.

3.3 Recursive implementation using an EKF

In order to demonstrate how the learning method can be implemented, an extended Kalman filter (EKF) is devised. It is one of the simplest ways to solve the optimization problem. The cost defined by Equation (13) is computed based on the current estimates and measurements, so the filter only stores the current parameters estimated values and covariance, computed under approximations of Gaussian noise and linearized models. More precisely, a variant of the EKF is used, which includes an exponential forgetting of past observations. The state transition and observation model are defined as

$$\theta_{k+1} = \theta_k + v_k, \;\; v_k \sim \mathcal{N}(0, Q_k) \tag{14}$$

$$\bar{z}_k = \bar{h}(\theta_k) + \bar{w}_k, \;\; \bar{w}_k \sim \mathcal{N}(0, \bar{R}_k) \tag{15}$$

where $\bar{z}_k$ is the observation vector z_k augmented with the *pseudo-measurement* associated to the constraint (5). This concept of *pseudo-measurements* has been used in the past to include equality constraints in state estimation schemes to improve tracking by explicitly including kinematic or dynamic constraints [10, 46]. The derivative of this constraint modeled by the *pseudo-measurement* is also used in order to exploit the joint velocity measurements $\dot{q}_{f,k}$ in addition to the joint positions $q_{f,k}$ such that

$$\bar{z}_k = \begin{bmatrix} \epsilon_{h,k} \\ \dot{\epsilon}_{h,k} \\ z_k \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ z_k \end{bmatrix} \tag{16}$$

$$\bar{h}(\theta_k) = \begin{bmatrix} \mathcal{K}(\theta_{r,k}, q_{f,k}) - g(\theta_{g,k}, t_k) \\ J_{\mathcal{K}}(\theta_{r,k}, q_{f,k})\dot{q}_{f,k} - \dot{g}(\theta_{g,k}, t_k) \\ h(\theta_k) \end{bmatrix} \tag{17}$$

$$J_{\mathcal{K}}(\theta_{r,k}, q_{f,k}) = \left.\frac{\partial \mathcal{K}(\theta_{r,k}, q_f)}{\partial q_f}\right|_{q_f = q_{f,k}} \tag{18}$$

and

$$\bar{R}_k = \begin{bmatrix} \Sigma_{h,k} & 0 & 0 \\ 0 & \dfrac{2}{T_s^2}\Sigma_{h,k} & 0 \\ 0 & 0 & R_k \end{bmatrix} \tag{19}$$

where the term $\frac{2}{T_s^2}\Sigma_{h,k}$ is the noise covariance matrix of $\dot{\epsilon}_{h,k}$. This expression arises from a naive time derivative of the noise model defined in Equation (10) that assumes a constant $\Sigma_{h,k}$, but in practice other expressions can be chosen. The parameters values and covariance estimates are computed with the EKF from this augmented observation using the update rule

$$\hat{\theta}_{k|k-1} = \hat{\theta}_{k-1} \tag{20}$$

$$\hat{P}_{k|k-1} = (1 + \alpha_k)\hat{P}_{k-1} + Q_k \tag{21}$$

$$\hat{\theta}_k = \hat{\theta}_{k|k-1} + K_k \left(\bar{z}_k - \bar{h}(\hat{\theta}_{k|k-1})\right) \tag{22}$$

$$\hat{P}_k = \hat{P}_{k|k-1} - K_k \bar{H}_k \hat{P}_{k|k-1} \tag{23}$$

where the subscript $k|k-1$ denotes a prior estimate at sample time k, $\alpha_k \geq 0$ is a factor tuning the weight given to past observations, K_k is the Kalman gain matrix such that

$$K_k = \hat{P}_k \bar{H}_k^T \left(\bar{H}_k \hat{P}_k \bar{H}_k^T + \bar{R}_k\right)^{-1} \tag{24}$$

and $\bar{H}_k$ is the Jacobian of the augmented observation model w.r.t. the parameters :

$$\bar{H}_k = \frac{\partial}{\partial\theta}\bar{h}(\theta)\bigg|_{\theta=\hat{\theta}_k, t=t_k} = \begin{bmatrix} -\frac{\partial}{\partial\theta_g} g(\theta_g, t) & \frac{\partial}{\partial\theta_r}\mathcal{K}(\theta_r, q_{f,k}) \\ -\frac{\partial}{\partial\theta_g}\dot{g}(\theta_g, t) & \frac{\partial}{\partial\theta_r}J_{\mathcal{K}}(\theta_{r,k}, q_{f,k})\dot{q}_{f,k} \\ \frac{\partial}{\partial\theta_g}h(\theta) & \frac{\partial}{\partial\theta_r}h(\theta) \end{bmatrix}_{\theta=\hat{\theta}_k,\, t=t_k}$$

This version of the EKF is often referred to as a fading memory Kalman filter [46] and the so-called fading factor α_k in Equation (21) allows to discard past observations. When $\alpha_k = 0$, the method is equivalent to a classical EKF. As the value of α_k increases, less weight is given to past observations.

4 APPLICATION OF THE METHOD

We propose a setup where an operator has to draw a path with a pen attached to the follower robot by moving a haptic interface used as leader robot and with the visual feedback of a camera. Though the correct path is known, its registration in the robot operational space is initially inaccurate. Additionally, the operator is free to impose the execution velocity such that the time parameterization of the task must be estimated online. Finally, the model of the complete kinematic chain including the robot and the pen is also partially inaccurate, with a parameter encoding the axial translation of the pen w.r.t. the end-effector has to be refined. Using the proposed method, the task and robot parameters are updated online from the teleoperation data and used to provide an adaptive haptic guidance to the operator.

4.1 Experimental setup

We implemented the position teleoperation using a Franka Emika Panda robot teleoperated by a Force Dimension Omega 7 haptic interface. Visual feedback was provided by a camera (webcam C930e, Logitech). The path to be drawn was printed on a tilted (30 deg.) planar support as depicted in Figure 3. In this way, the position of the support in the horizontal plane of the world frame and its orientation around the vertical axis affect the task along all three dimensions.

The follower robot end effector was equipped with a force/torque sensor (nano 43, ATI), a pen-holder, and a pen (see Figure 3.B). An impedance controller tracks the pen-holder Cartesian pose reference imposed by the operator at the leader station (redundancy and inverse kinematics are handled by the controller). The end-effector orientation is constant and computed such that the pen remains normal to the drawing plane. The remaining 3 DOF that correspond to the position of the pen tip are controlled to perform the drawing task. Since the orientation of the follower robot is fixed, the leader robot is used as a 3 DOF device and its orientation is ignored. A controlled constant force of 1 N is applied to the pen to guarantee permanent contact with the surface, which leaves 2 DOF for the operator to control. The force control normal to the surface limits the cognitive load that more complex force feedback (even admittance/impedance control) could generate in the presence of haptic guidance. It also prevents possible interferences between force feedback and haptic guidance that can sometimes compensate one another [47]. Although the operator can only control 2 DOF of the follower robot, the trajectory to be performed on the leader side, as well as the provided haptic guidance, are three-dimensional. The end-effector pose of the robot $x_{ee,k}$ corresponds to the position of a point attached to the pen-holder, i.e., to the base of the pen. Besides, $x_{f,k}$ is the position of the tip of the pen actually used to perform the drawing task.

In the proposed experimental setup, a tool grasping model defines the relationship between the position of the end effector $x_{ee,k}$ and that of the tip of the tool $x_{f,k}$, such that the transformation between $x_{ee,k}$ and $x_{f,k}$ is a translation along the Z axis of the end-effector frame. The robot model parameter $\theta_{r,k} = L_k$ is the distance

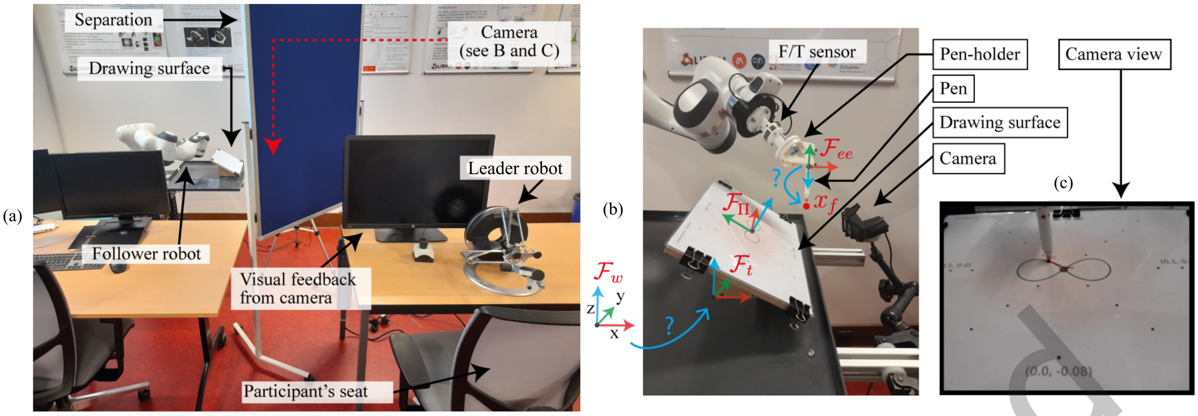

Fig. 3. (a) Experimental setup. (b) Close view of the task support, the robot end-effector, and the pen. The main frames of reference used in the following sections are represented. $\mathcal{F}_{\Pi}$ is an additional frame aligned with the drawing plane used for convenience when displaying results. The frame attached to the tip of the pen is omitted for clarity. (c) Visual feedback from the camera available to the human operator to perform the teleoperation task.

between the pen holder position and the tip of the pen. The tool registration model is combined with the known forward kinematic model of the Panda robot to obtain the follower robot model $x_{f,k} = \mathcal{K}(\theta_{r,k}, q_{f,k})$. This scenario is similar to what could happen in another context if a gripper was used to hold a tool: the exact tool configuration w.r.t. the end effector could be only partially known.

Let $\Gamma(\psi) \in \mathbb{R}^3$ be a smooth path parameterized by its curvilinear abscissa ψ. The path is planned in a planning frame $\mathcal{F}_t$ and subsequently registered in the world frame $\mathcal{F}_w$. In the experiment, the rigid transform from $\mathcal{F}_w$ to $\mathcal{F}_t$ results from the combination of translations along X and Y axes, respectively denoted as $t_{x,k}$ and $t_{y,k}$, and a rotation about Z axis with angle $r_{z,k}$ (see Figure 3.b). The time parameterization of the task is modeled by an affine function of time, such that $\psi(t_k) = a_k + b_k\, t_k$. The task model is then:

$$x^d_{f,k} = g(\theta_{g,k}, t_k) = \begin{bmatrix} t_{x,k} \\ t_{y,k} \\ 0 \end{bmatrix} + {}^{w}R_t(r_{z,k})\Gamma(a_k + b_k t_k) \tag{25}$$

where $\theta_{g,k} = \left[a_k, b_k, r_{z,k}, t_{x,k}, t_{y,k}\right]^T$ is the set of task parameters and ${}^{w}R_t(r_z) \in SO^3$ is the rotation matrix between frames $\mathcal{F}_w$ and $\mathcal{F}_t$. The task parameters to be refined are therefore the position and orientation of the support in $\mathcal{F}_w$ and the two time parameters (i.e., 5 parameters in total). $\mathcal{F}_{\Pi}$ (see Figure 3.b) is an additional frame aligned with the drawing plane used for convenience when displaying results in the following.

The ROS2 framework was used to handle the different multi-rate components (see Figure 4). Haptic guidance was computed according to Section 2.3 and implemented using the Force Dimension SDK. The stiffness of the haptic guidance is set to 300 N.m^{-1} and the damping is chosen so that the equivalent system is a damped mass-spring system with a damping coefficient of 0.9. The learning process takes as inputs both the current execution time and the follower robot measurements with sampling period $T_s = 0.02$ s and sends the current task and robot parameter estimates to the high-level controller as detailed in Figure 4. This high-level controller computes the guidance reference and the follower robot reference from, respectively, the estimated task model and the estimated tool registration model. It also manages the position-position mapping between leader and follower. This architecture is implemented on a Linux platform (i7 / 32GB ram).

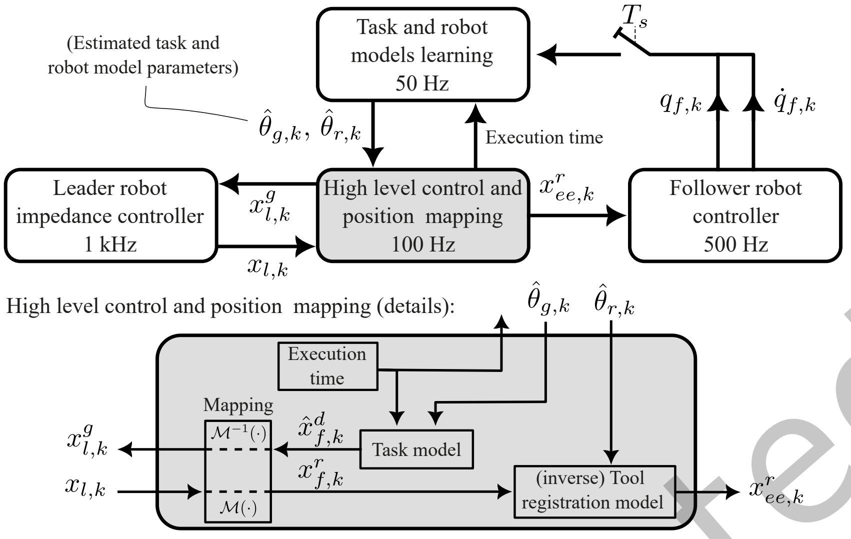
(Estimated task and robot model parameters)
Task and robot models learning 50 Hz
T_s
$q_{f,k}$
$\dot{q}_{f,k}$
$\hat{\theta}_{g,k}, \hat{\theta}_{r,k}$
Execution time
Leader robot impedance controller 1 kHz
$x^g_{l,k}$
$x_{l,k}$
High level control and position mapping 100 Hz
$x^r_{ee,k}$
Follower robot controller 500 Hz
High level control and position mapping (details):
$\hat{\theta}_{g,k}$
$\hat{\theta}_{r,k}$
Execution time
Mapping
$\mathcal{M}^{-1}(\cdot)$
$\mathcal{M}(\cdot)$
$\hat{x}^d_{f,k}$
$x^r_{f,k}$
Task model
(inverse) Tool registration model
$x^g_{l,k}$
$x_{l,k}$
$x^r_{ee,k}$

Fig. 4. Overview of the software architecture and main variables. Details are provided for the high-level controller handling the mapping from $\mathcal{F}_l$ to $\mathcal{F}_w$, the generation of the guidance reference $x^g_{l,k}$, and the generation of the follower robot end-effector Cartesian reference $x^r_{ee,k}$.

4.2 Choice of the learning hyper-parameters

Hyper-parameters have to be tuned in order to reach satisfactory learning performances. Some general insights are provided thereafter to illustrate their effect on the learning performance.

The fading factor α_k allows to tune the weight given to past observations such that their effect becomes negligible after a certain time. This allows to cope with potential parameter drift and modeling errors arising from the linearization of the possibly non linear models [45]. Large values of α_k improve the learning performance, but reduce the robustness to execution errors. Indeed, if all past data are discarded, a local deviation from the desired trajectory (i.e., execution errors) will cause the estimated parameters $\hat{\theta}_k$ to degrade immediately. In the experiments, the fading factor is set to a constant value $\alpha_k = \alpha = 1.10^{-3}$.

An intuitive way to characterize the exponential fading memory effect over time is to consider the relative weight given to past observations. The decay time constant τ of this weight is

$$\tau = \frac{T_s}{\ln(1+\alpha)} \tag{26}$$

Equivalently, Equation (26) can be rearranged as $\alpha = e^{T_s/\tau} - 1$ and using this expression, a value of α can be computed from the desired decay time constant τ [26]. The decay time constant associated with $\alpha = 1.10^{-3}$ would therefore be $\tau = 20$s. This means that after one minute, the weight given to an observation has virtually decreased by 95%. It is a good choice if the parameters do not vary much, because the errors introduced by the human operator are effectively filtered out over a span of nearly one minute. Results to illustrate the effect of α are reported in Section 5.4.

The human execution covariance Σ_h should be chosen to model the human behavior and reflect the expected execution errors considering the sampling period. Generally, a larger covariance matrix slows down the learning by decreasing the weight given to the demonstrations. Slightly overestimating the execution errors allows to smooth the parameters estimation to obtain a more gradual learning, as chosen for the following experiments. Similarly, the initial estimated parameter covariance P_0 should encode the real uncertainty over parameters estimates to reach an optimal performance, but most often this information is unknown. Then, the initial covariance should be underestimated to avoid sudden updates of the parameters at startup when the information coming from the user is often the least reliable. The transient behavior at startup only depends on the chosen values for P_0 and Σ_h and the proposed implementation with the fading memory limits the influence of inaccurate covariance modeling over time. In the following experiments, the covariance matrix of the execution error introduced by the operator is set to $\Sigma_{h,k} = \sigma_h^2 I$, where $\sigma_h = 5$ mm and P_o is chosen as a diagonal matrix modeling initially independent estimated parameters of standard deviations equal to 10 mm for $\hat{a}$, 10 mm.s^{-1} for $\hat{b}$, 1 deg. for $\hat{r}_z$, and 1 mm for $\hat{t}_x$, $\hat{t}_y$ and $\hat{L}$.

The model of $\psi(t)$ is learned online to accommodate for the velocity and delays imposed by the operator. Nevertheless the model may become inaccurate when the operator is slowing down, delaying before starting the gesture, or pausing during the task execution. The estimation error on the associated parameters may propagate to the other parameters. In order to ensure that the tracking of ψ is reactive, the possible change in pace is modeled with the process noise variance of $\dot{\psi}$ set to $\sigma_{\dot{\psi}} = 0.4$ mm.s^{-1}. This results in additional terms in the process noise covariance matrix Q_k that only impact the process noise covariance associated to the time parameters a and b. The chosen time parameterization is of the form $\psi(t) = a + bt$ such that

$$\begin{bmatrix} \psi \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \tag{27}$$

The modeled noise of standard deviation $\sigma_{\dot{\psi}}$ on $\dot{\psi}$ at time t_k can then be propagated to the time parameters using Equation 27. This results in a process noise covariance matrix of the form

$$Q_k = \begin{bmatrix} Q_{(a,b),k} & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{28}$$

where

$$Q_{(a,b),k} = \begin{bmatrix} 1 & -t_k \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & \sigma_{\dot{\psi}}^2 \end{bmatrix} \begin{bmatrix} 1 & -t_k \\ 0 & 1 \end{bmatrix}^T \tag{29}$$

5 EXPERIMENTAL RESULTS

The performance of the learning method is demonstrated on a task consisting in drawing a treble key. An operator manipulates the haptic interface to follow the path printed on the surface with the pen attached to the follower robot. The estimated task and robot models, and consequently the generated haptic guidance, are initially incorrect.

5.1 Trajectories evaluation

The ground truth path is given by the template printed treble key. Since the pace of the realization is not imposed, the time-stamping needed to define the desired trajectory $x_{f,k}^d$ is determined once the task execution is over. For all time steps, $x_{f,k}^d$ is defined as the point on the ground truth path closest from the actual follower position $x_{f,k}$. This computation is necessary to account for different execution paces, because, while the spatial registration can be calibrated beforehand, the correct time registration is the one imposed by the human operator during the task execution. The execution error is defined at each time step as the deviation from the correct path and computed

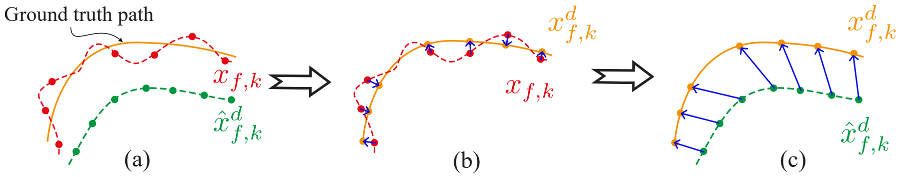

Fig. 5. (a) Illustration of the collected trajectories $x_{f,k}$ and $\hat{x}^d_{f,k}$. (b) The desired trajectory $x^d_{f,k}$ is computed by finding on the ground truth path the points closest to $x_{f,k}$ (for each sample), which then results in errors $||\epsilon_{h,k}||$ (see b) and $||x^d_{f,k} - \hat{x}^d_{f,k}||$ (see c).

as $||\epsilon_{h,k}||$ where $||\cdot||$ denotes the Euclidean norm. The task model prediction error at each time step is computed as $||\tilde{x}^d_{f,k}|| = ||x^d_{f,k} - \hat{x}^d_{f,k}||$. Similarly, the robot model prediction error is computed as $||\tilde{x}_{f,k}|| = ||x_{f,k} - \hat{x}_{f,k}||$. Note that, in the present experiment, L which encodes the length of the pen is the only initially unknown robot model parameter. Therefore the robot model estimation error is simply $||x_{f,k} - \hat{x}_{f,k}|| = \tilde{L}_k$.

In order to capture overall trends in model parameter estimation errors, the mean relative parameter error w.r.t. the initial errors is computed as:

$$\tilde{\theta}_{\text{rel.},k} = \frac{1}{4}\left(\left|\frac{\tilde{r}_{z,k}}{\tilde{r}_{z,0}}\right| + \left|\frac{\tilde{t}_{x,k}}{\tilde{t}_{x,0}}\right| + \left|\frac{\tilde{t}_{y,k}}{\tilde{t}_{y,0}}\right| + \left|\frac{\tilde{L}_k}{\tilde{L}_0}\right|\right) \tag{30}$$

At startup $\tilde{\theta}_{\text{rel.},k} = 1$ and $\tilde{\theta}_{\text{rel.},k} = 0$ would correspond to the ideal convergence of estimation errors to zero. The parameter errors on a_k and b_k are not included in Equation (30), since they tend to vary locally as the velocity imposed by the operator changes.

In addition, in order to assess how the learning method would perform in the most favorable scenario, the learning performance is also assessed for a perfect execution of the task at a constant pace. Since this cannot be achieved by a human operator, we use an automatic robotic execution. To do so, the end-effector reference $x^r_{ee,k}$ sent to the robot controller is computed using the correct models, estimated from offline registration and tool calibration. This condition , which does not include an operator in-the-loop, is referred to as **AUTO**, whereas the proposed method with an operator in the loop is referred to as **AG** (Adaptive Guidance).

5.2 Evolution of the estimated desired trajectory

In Figure 6, the paths underlying the desired trajectory $x^d_{f,k}$ and its estimation $\hat{x}^d_{f,k}$ acquired during an **AG** scenario are displayed in $\mathcal{F}_\Pi$ for different times t_k. At startup, $\hat{x}^d_{f,k}$ is significantly different from $x^d_{f,k}$ due to the parameter estimation errors and the operator is therefore initially guided along an incorrect trajectory. The drawing plane being at an angle w.r.t. the horizontal plane in $\mathcal{F}_w$, the incorrect estimation of the orientation r_z impacts not only the in-plane orientation of the estimated desired trajectory, but also the predicted desired position along the normal to the plane. This can be observed in Figure 6 where the position and orientation of the initial estimated task model projected in the XZ plane are incorrect. Despite the guidance errors, the operator is able to perform the desired task, although execution errors are introduced.

5.3 Quantitative results

Three arbitrary initial task registration errors have been considered. The results are reported in Figure 7 and 8. For the three **AG** executions, the task execution error is maximum at startup and can reach 7 mm. It is 0.62 ± 0.09 mm on average with local maxima at 2.5 mm (see Figure 7). As the operator performs the task, the models are refined

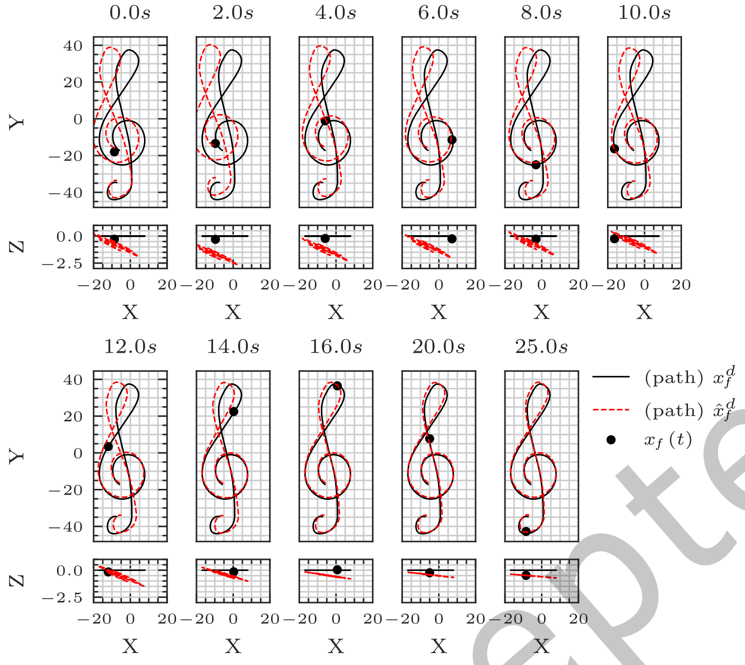

Fig. 6. Estimated task model evolution over time for the scenario **AG** with initial parameters estimation errors $\tilde{r}_{z,0} = 10$ deg., $\tilde{t}_{x,0} = 2$ mm, $\tilde{t}_{y,0} = 5$ mm, and $\tilde{\theta}_{r,0} = \tilde{L}_0 = 10$ mm. The task model is represented as the path underlying $\hat{x}^d_{f,k}$ and all units for X, Y, and Z are in mm.

by the learning process and the estimated trajectory $\hat{x}^d_{f,k}$ gradually converges towards $x^d_{f,k}$as shown by the task prediction error (see Figure 7), which confirms observations made in Figure 6. The estimated robot position $\hat{x}_{f,k}$, initially incorrect by 10 mm along the Z axis of $\mathcal{F}_\Pi$ also converges towards its ground truth value as the estimation of L is updated (see Figure 8).

The evolution of $\tilde{\theta}_{\text{rel.},k}$ reported in Figure 8 (bottom) shows that parameter estimation errors are largely reduced (by 85% or more) over the task duration. The individual parameter estimation error profiles are similar for **AUTO** and **AG**, which suggests that the learning method can effectively cope with noisy human demonstrations. Even if the human demonstrations lead to learning performances inferior to those obtained under the **AUTO** scenario, the task prediction errors under **AG** are also reduced by more than 75%. It can be observed that the steady state has not been reached because of the limited execution time. This will be discussed in section 5.4. An additional reason is linked to the geometry of the task can prevent the full convergence of some parameters. For instance, the downward motion from the top of the key up to the bottom (see Figure 6) provides very little information about the translation of the task along this direction and the associated task translation parameter $\hat{t}_{x,k}$ is barely updated over this time range (see for instance the orange curve in Figure 8 for $t_k \in [18; 23]$ s).

The same experiment was carried out with a variety of initial parameter estimation errors as well as with another underlying path $\Gamma(\psi)$, i.e., the treble key (see Figure 6) and the lemniscates presented in Sections 5.5 and 5.6 (see Figures 10 and 11, respectively). The learning method was found to be task-independent in the sense that the hyper-parameters (R_k, Q_k, P_0, and α) can be kept constant from one experiment to another while providing satisfactory results.

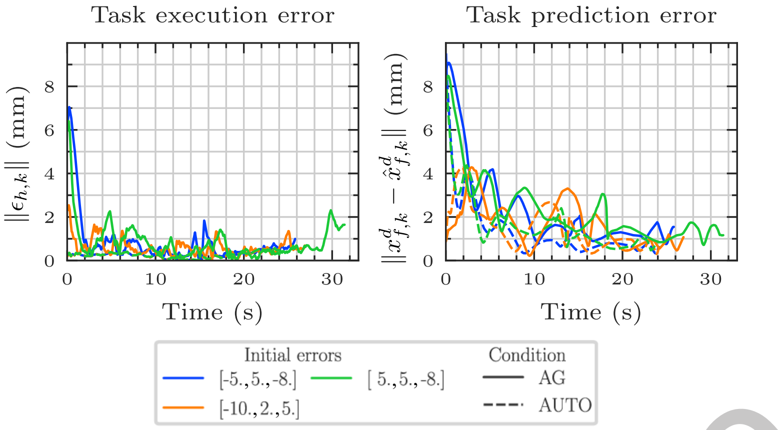

Fig. 7. The norm of the execution (left) and prediction (right) errors under **AG** and **AUTO** are reported for different initial task parameter estimation errors $[\tilde{r}_{z,0},\ \tilde{t}_{x,0},\ \tilde{t}_{y,0}]$ in [deg., mm, mm]. The initial robot parameter estimation error is $\tilde{L}_0 = 10$ mm.

5.4 Influence of the fading factor

The decay time constant (20 s) associated with the chosen value for α is rather slow considering the total duration of the experiment presented above ($\approx$ 30 s), which accounts for part of the residual errors. To demonstrate the influence of α on the learning, the **AUTO** condition is executed for various values of the fading factor and the evolution of $\tilde{\theta}_{\text{rel.}}$ is reported in Figure 9. The estimated parameters converge faster towards their correct values for larger values of α. Indeed, the estimated covariance of the parameters is greater due to the forgetting mechanism, leading to a better learning performance when the observations are correct. For $\alpha = 0.01$, equivalent to a decay time-constant of 2 s, only the last few seconds are considered for learning (see Equation 26).

5.5 Robustness to parameters shift

One of the advantages of the presented general method and its implementation with a fading memory EKF is the possibility to cope with parameters shift. As a demonstration, let us consider a scenario where the participant is asked, under the **AG** conditions, to follow a closed path A (lemniscate) for one revolution and then to follow a path B generated with different parameters (translation w.r.t. world frame) for the second revolution, which effectively displaces the desired path. Note that experimental conditions and hyper-parameter values are the same as for section 5.1. In this scenario, our method can cope with small parameters shift as demonstrated by Figure 10. The task model is able to adapt to the change of path without requiring any hyper-parameter tuning. It should be noted that the general formulation of the learning method (see Section 3) assumes that the parameters are constant. The process noise included in the model by Equation 14, allows to account for slowing varying parameters only. The ability to cope with abruptly changing parameters shown in this section is rather linked to the robustness of the Kalman filtering estimation method.

5.6 Necessity for joint robot and task models corrections

In order to exhibit the need to correct the follower robot kinematic model in addition to the task model, we carried out experiments with the lemniscate path without updating the robot model parameter (i.e., θ_r is an empty set). The results under this scenario **AG-TO** (AG Task Only) are reported in Fig.11, where the predicted desired trajectory, displayed as a path in the XY and XZ drawing planes, clearly does not converge towards the

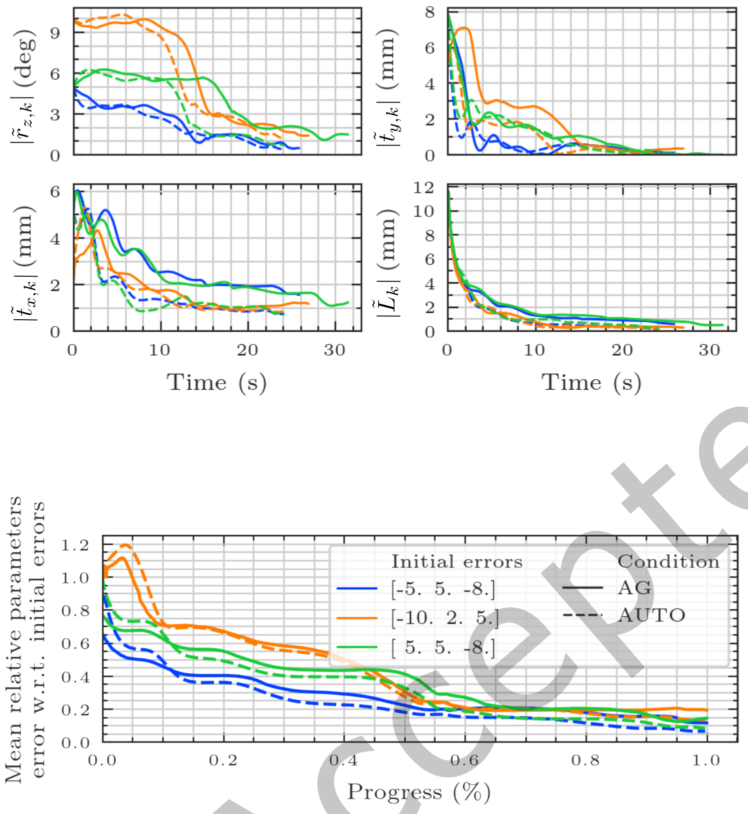

Fig. 8. Top: parameters absolute estimation errors under **AG** and **AUTO** for different initial task parameter estimation errors $[\tilde{r}_{z,0},\ \tilde{t}_{x,0},\ \tilde{t}_{y,0}]$ in [deg., mm, mm]. The initial robot parameter estimation error is $\tilde{\theta}_{r,0} = \tilde{L}_0 = 10$ mm. Bottom: resulting mean relative errors $\tilde{\theta}_{\text{rel.},k}$ defined by Equation (30).

actual desired trajectory. Some of the underlying parameters, namely the orientation $\hat{r}_z$ and lateral translation $\hat{t}_y$, partially converge towards the correct values, but large errors remain for the other parameters (see Fig.12), notably the estimation of t_x that converges towards a wrong value. Similar results are obtained when learning from an optimal demonstration (**AUTO-TO**) (see Fig.12).

6 COMPARISON WITH NON-ADAPTIVE GUIDANCE

In the previous section, we showed that user inputs can be used to simultaneously improve the task and robot models when they are initially incorrect. The online learning method was used to provide an adaptive guidance to the human operator. To further demonstrate the advantages of this approach, the task performance is evaluated in four cases: without haptic guidance, with an adaptive guidance, with an incorrect non-adaptive guidance and with a correct one.

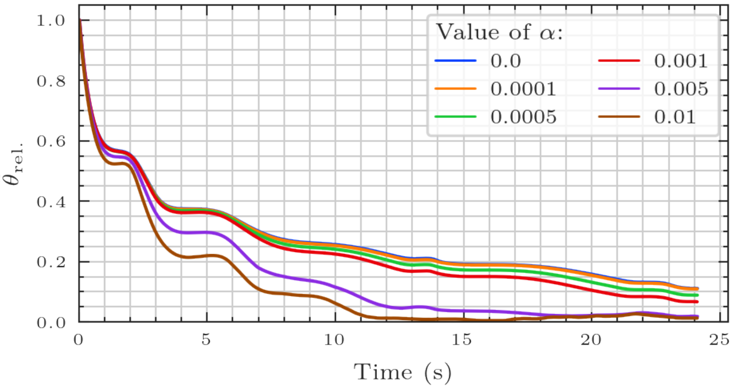

Fig. 9. Average parameter estimation errors as a fraction of its initial value for different values of the fading factor α. The data was collected from 6 optimal executions, i.e., the **AUTO** condition.

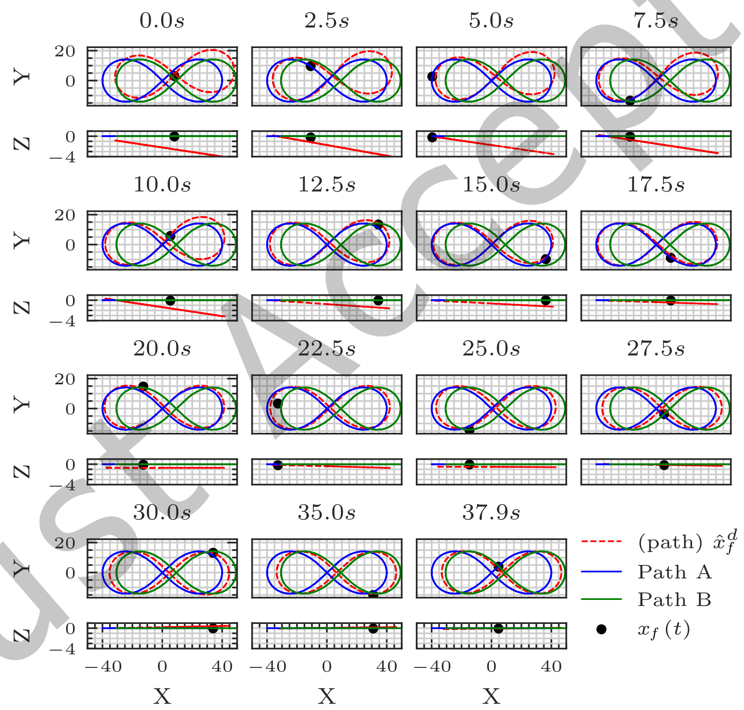

Fig. 10. Illustrative scenario simulating a parameters shift when the first revolution is completed (around t_k = 18 s). Two paths A and B are visible on the drawing surface such that path B is identical to path A, but translated by 10 mm along the X axis of the drawing plane. The participant follows the path A for one revolution and then path B. The task model initially converges towards path A, but is quickly adapted when a new path is followed. All distances are expressed in mm.

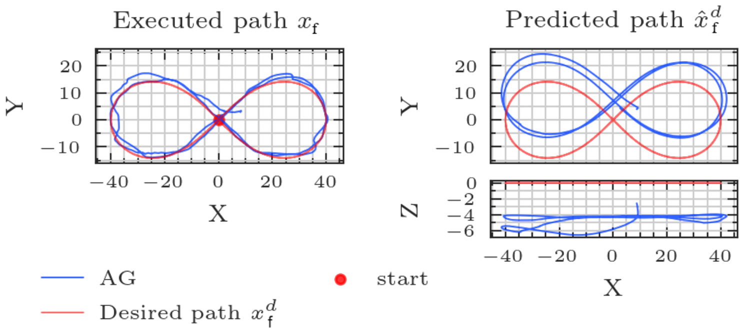

Fig. 11. Executed, desired, and predicted trajectories displayed as paths for the **AG-TO** scenario. All units for X, Y, and Z are in mm

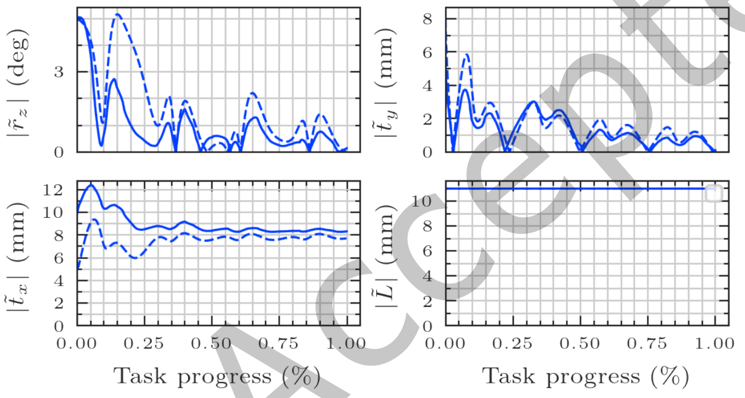

Fig. 12. Parameter estimation errors for the **AG-TO** (solid lines) and **AUTO-TO** (dotted lines) scenarios.

6.1 Experimental setup

The context is the same as the one detailed in Section 5.1: the model of the path to follow is known, but its registration is partially unknown, and the tool grasping configuration is also unknown. Initially, only rough estimates of the parameters θ_g and θ_r are available.

The **AG** condition provides haptic guidance and learning of the unknown task and robot parameters. In order to assess the effect of model inaccuracies and online corrections, three additional experimental conditions are considered:

- **NG** – no guidance forces are applied on the leader robot;
- **IG** – A (non-adaptive) guidance is provided, which uses the initial incorrect model parameters;
- **CG** – A (non-adaptive) guidance is provided for which the model parameters are correct.

Under conditions **CG** and **IG**, only the tracking of the parameters a and b is performed to accommodate the execution velocity imposed by the operator. The other four parameters, encoding the task and tool partial registration, are not updated such that $\theta_k = \theta_{g,k} = [a_k, b_k]^T \in \mathbb{R}^2$. This condition can be considered as a normal virtual fixture and behaves as existing (non-adaptive) haptic guidance methods, such as Guidance VF [7]. The only particularity is that, due to the proposed method, the guidance reference is not simply computed as the closest point on the guidance path, but from the estimated parameters a_k and b_k. This tracking introduces dynamics in the VF that is experienced by the operator as inertia in the direction tangent to the guidance path, which tends to smooth the execution by filtering accelerations. For **CG** the registration of the desired path is exact, while for **IG** the registration is inaccurate. Under the two other conditions **AG** and **NG**, the proposed online learning method is used to update the initially incorrect parameters from the observed user actions. What differentiates **AG** and **NG** is how the models are used. The former provides a haptic guidance whose underlying model is refined as the parameters are updated whereas the later passively observes without providing any haptic guidance. The learning performance for six correct automatic executions (i.e., **AUTO** condition) was also added to provide a baseline performance.

6.1.1 Parameters tuning. The haptic guidance is implemented with a stiffness of 300 N.m^{-1} and a damping ratio of 0.9. The fading memory EKF is implemented with a constant fading factor $\alpha = 10^{-3}$ (20 s decay time constant) and an isotropic Gaussian model of standard deviation $\sigma_h = 2$ mm for the modeled human inaccuracies. The noise modeled on the execution velocity imposed by the operator is $\sigma_{\dot{\psi}} = 0.1$mm.s^{-1}. The initial parameter covariance matrix is initialized with small values encoding an initial standard deviation of 10 mm for $\hat{a}$, 10 mm.s^{-1} for $\hat{b}$, 1 deg. for the in-plane task orientation, and 1 mm for the three remaining task and robot parameters.

6.1.2 Clutching for mapping recalibration. The considered task shown in Figure 14 spans a larger area than for the previously reported experiments. In order for the operator to be able to execute a task spanning a significant part of the follower robot operational space, mapping clutching [28] was implemented.

In practice, the mapping $\mathcal{M}(\cdot)$ was modified such that $\mathcal{M}(x_l)$ stays constant during the so-called clutching phases (i.e., when a button is pressed), therefore generating no follower robot displacement. The operator can then manually adapt the mapping between leader and follower position by repositioning the leader independently from the follower when clutched.

6.2 Experimental protocol

Twelve participants, all right handed, were recruited as part of a study approved by an ethical committee [1]. After being given adequate information about the material and asked to fill the consent forms, each participant practiced 5 minutes with and without haptic guidance by drawing a circle. They then performed a wide motion (e.g., line) to practice clutching and finally performed free form drawings. Once the training phase was finished, each participant performed an imposed task (see Figure 14) under the 4 experimental conditions (**AG**, **CG**, **IG**,**NG**). The participants were blinded to the conditions, which were presented in a pseudo-random balanced order.

For each realization the user started the task by pressing a push button (also used for clutching) on the haptic interface, that engages the position control. The robot is then brought to the startup position, and then the path is followed under visual feedback (see Figure 13).

After completing each task under a particular condition, each participant filled a questionnaire to evaluate subjective preferences (see section 6.2.1 for details).

Throughout the experiments, we tested our main hypotheses that can be stated as:

- **H1** – when haptic guidance is provided, if the models of the task and/or of the robot are inaccurate, the execution errors will be similar or larger than if no guidance was provided;
- **H2.1** – although imperfect, human actions can be used to learn the correct task and robot models;
- **H2.2** – by correcting the models online, conflicts are significantly reduced, which leads to better performances overall in comparison to a non-adaptive guidance.

[1] University of Strasbourg, France. Accreditation n° **Unistra/CER/2022-09**.

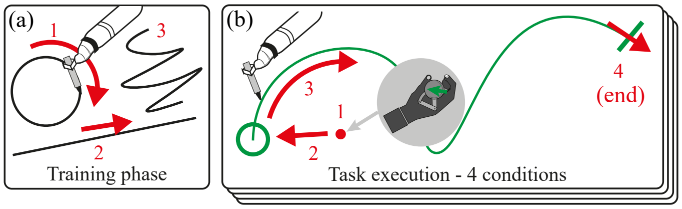

Fig. 13. Overview of experimental protocol. (a) Participants are asked to practice 5 minutes with and without haptic guidance. (b) Then, they perform the same task under 4 experimental conditions and fill a questionnaire at the end of each condition.

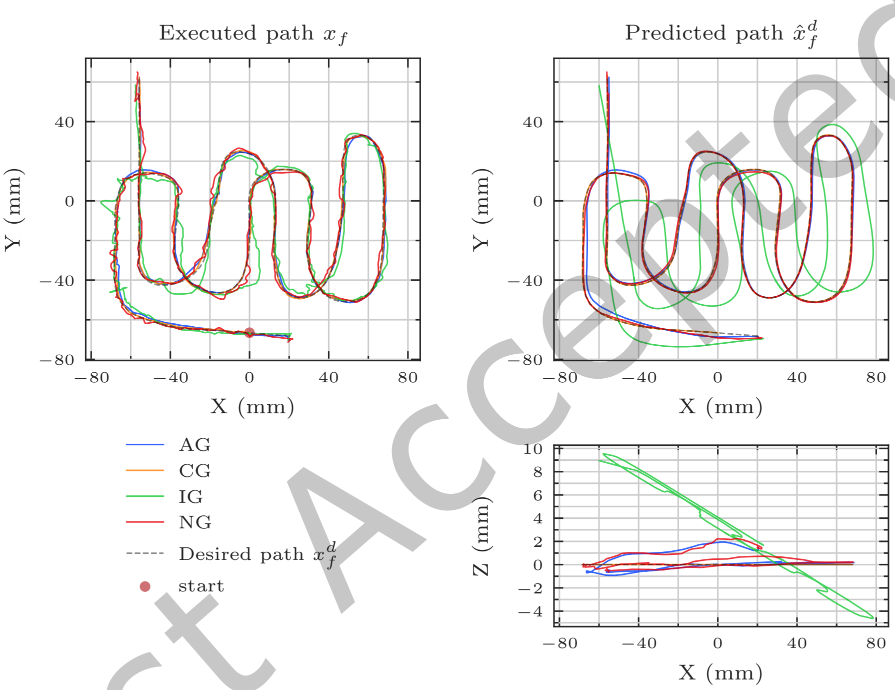

Fig. 14. Path underlying the trajectories $x_{f,k}$ and $\hat{x}_{f,k}$, respectively the executed and predicted trajectories, expressed in $\mathcal{F}_t$ for one of the participants under the different experimental conditions. See figure Appendix 18 for snapshots at different time points (**AG** condition).

6.2.1 Collected data. The executed ($x_{f,k}$) and predicted ($\hat{x}^d_{f,k}$) trajectories are recorded and compared with the ground truth task model (i.e., desired trajectory $x^d_{f,k}$) to compute the execution errors $||\epsilon_{h,k}||$ and task prediction errors, respectively. Likewise, the robot estimated trajectory using the robot model $\hat{x}_{f,k}$ is recorded and compared with executed trajectory $x_{f,k}$ to compute the robot prediction errors. Additionally, the total execution time (task duration), is recorded.

The smoothness of the executed follower robot trajectory is also evaluated using the so-called spectral arc-length metric [3]. The spectral arc length of the follower Cartesian velocity profile was computed using the

SPARC library [3]. The method was implemented with a cut-off frequency of 10 Hz and a threshold of 5.10^{-2} for signal amplitude.

In addition to objective measurements, subjective assessments from the users were collected. After completing each task under a particular condition, each participant filled a NASA-TLX [14] questionnaire. In addition to the NASA-TLX questionnaire, two questions were included to evaluate if the participants found the haptic guidance to be useful (see Table 1). As the second question (Q2) requires the haptic guidance to be activated, it was ignored when no force was applied (**NG**).

	Question	Likert scale (5 points)
Q1	I felt in control while performing the task.	1: Strongly disagree 4: Agree
Q2	The forces generated by the leader interface helped me to perform the task.	2: Disagree 5: Strongly agree 3: Neither agree nor disagree

Table 1. Additional survey questions for each condition. Translated from French.

6.3 Results

The collected experimental data for one participant under the 4 conditions is displayed in Figure 14 for illustration. Under **IG** and **AG**, the task model registration is incorrect when the task execution starts. It is refined under **AG**, such that the task model becomes correct after 20 to 30s (see snapshots in Appendix 18), which is between 15% to 25% of the average total task duration.

6.3.1 General behaviors for the different guidances. In order to describe the typical behaviors obtained with the different compared guidance methods, we first observe and compare the realizations by one particular participant reported in Figure 14. In the case of correct guidance (**CG**), the user just has to follow the guidance provided by the haptic interface, which allows to obtain a smooth path (shown in orange on the left figure) very close to the desired trajectory (shown with the dashed line). This very good behavior is obtained because the path predicted by the system shown in orange on the right figure superimposes perfectly with the desired path (shown with dashed line). This can, however, only be obtained if both the task model and the robot model are correct. On the contrary, in the case of incorrect guidance (**IG**) the path predicted by the system shown in green on the right figure differs significantly from the actual desired path both in the X-Y plane and along the Z-axis. In the case of this non-adaptive haptic guidance, the parameters of the task model and of the robot model are not updated during the realization of the task by the user. Because of the discrepancy between predicted and desired path, the user has to constantly apply forces against the (incorrect) haptic guidance to be able to follow the desired path. While this is possible because the haptic forces provided by the interface are limited, this is a difficult task, which results in the actual path (shown in green in the left figure), which coarsely remains around the desired path, but which is imprecise. Large errors can in particular appear when the direction of the guidance force that the user has to fight abruptly changes. This can be seen around position $X = 0$mm, $Y = -40$mm, where the erroneous guidance force changes from the Y direction to the X direction because of the 90° turn in the trajectory, hence creating the overshoot along X. This kind of behavior would also be obtained in the case of a static virtual fixture if the robot model and / or the task model are incorrect. In the case of no guidance (**NG**), the user does not have to fight against haptic guidance, but, without assistance, the effects of tremors or badly controlled motions can appear all over the path, as shown by the red path on the left figure, which slightly oscillates around the desired path.

Finally, in the case of Adaptive Guidance (**AG**), the predicted path shown in blue in the right figure is originally incorrect and deviates from the desired path (see in particular the Z-axis). However, the efforts of the user against the originally incorrect guidance are used to update the task and robot model using the proposed approach, so that after a few curves (around $X = -40$ mm, $Y = -20$ mm), the predicted path almost perfectly superimposes with the desired path. As a consequence, the user receives a good guidance, which allows him/her to perform the end of the task with a good accuracy. Note that these observations are provided for bringing a general overview of the behaviors for the different guidance conditions. They are, however, based on a specific case and the behaviors can slightly vary depending on the ability of the user to feel haptic guidance and to move against incorrect haptic guidance. The full results will be statistically analyzed and discussed in the next sections.

6.3.2 Objective results across participants. An overview of the objective results from our user study is reported in Table 2. A repeated measures analysis of variance (ANOVA) was performed with the experimental condition as factor. Note that the **AUTO** condition is excluded from ANOVA and subsequent pairwise post hoc comparisons as it is not performed by the participants. We found significant effects of the conditions on mean execution errors ($F(3, 33) = 31.17, p < 1.10^{-3}$), task model prediction errors ($F(3, 33) = 5929.32, p < 1.10^{-3}$) and robot model prediction errors ($F(3, 33) = 112.75, p < 1.10^{-3}$). However, the total task duration (about 2 minutes on average) was not found to be significantly impacted by the experimental condition ($F(3, 33) = 3.26, p = 1.10^{-1}$). Post hoc analysis of the pairwise comparisons are reported in Figures 15 using asterisks to denote statistical significance.

The main significant results are as follows:

- task execution errors under **CG** and **AG** are significantly lower than under **NG** and **IG** ($p < 1.10^{-2}$ for all comparisons);
- task prediction errors are significantly smaller under **CG** than under **AG** ($p < 1.10^{-4}$), **NG** ($p < 1.10^{-3}$), or **IG** ($p < 1.10^{-4}$) and the same goes for robot model prediction errors ($p < 1.10^{-2}$ for all comparisons);
- task prediction errors are significantly reduced by the learning process (**AG** and **NG** compared to **IG**, $p < 1.10^{-4}$ for both);
- robot prediction errors are also significantly reduced by the learning process (**AG** and **NG** compared to **IG**, $p < 1.10^{-2}$ for both);
- there is no statistically significant difference between the prediction errors, both for task and robot, under **AG** and **NG**.

Concerning the smoothness, as would be expected, the automatically generated motion (**auto**) is much smoother than the ones performed by the participants. For those, the experimental condition was found to have a significant effect on the smoothness ($F(3, 33) = 13.59, p < 1.10^{-4}$) . The results show that the motion is significantly less smooth under **NG** than under any other condition (see Figure 15). The only other significant result is that the motion is significantly smoother under **AG** than under **IG** ($p < 5.10^{-2}$). It can also be noted that **AG** and **CG** have comparable scores ($p = 0.91$).

Metric	AG	CG	IG	NG	AUTO
Task duration (s)	119.34 ± 34.63	121.50 ± 37.31	134.54 ± 39.46	133.00 ± 33.80	87.04 ± 0.02
Execution errors (mm)	0.90 ± 0.21	0.82 ± 0.19	1.64 ± 0.49	1.35 ± 0.35	0.13 ± 0.05
Task model pred. errors (mm)	1.06 ± 0.27	0.42 ± 0.21	10.24 ± 0.16	0.99 ± 0.30	0.23 ± 0.00
Robot model pred. errors (mm)	0.57 ± 0.25	0.05 ± 0.02	11.33 ± 0.49	0.44 ± 0.1	0.18 ± 0.06
Smoothness (unitless)	−4.13 ± 0.47	−4.30 ± 0.91	−5.01 ± 0.77	−6.39 ± 1.52	−2.40 ± 0.02

Table 2. Objective performance metrics mean values and standard deviations across participants ($N = 12$).

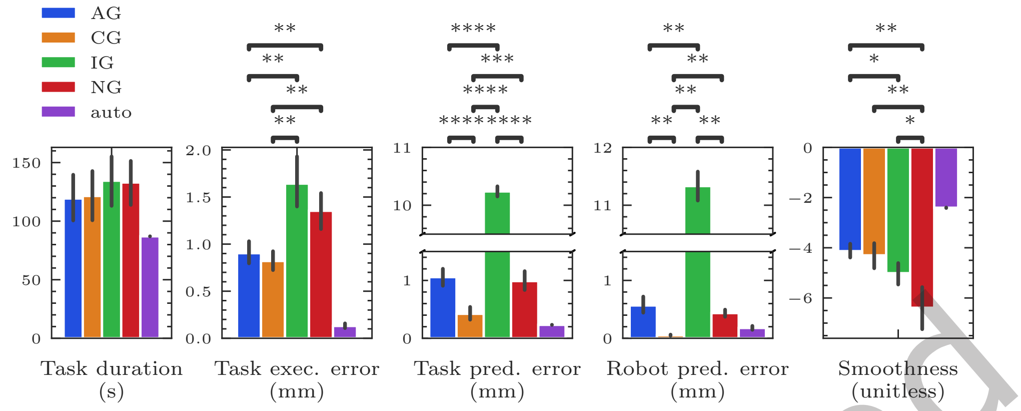

Fig. 15. Mean task duration, execution errors (path deviation), task and robot model prediction errors, and smoothness across participants ($N = 12$). The smoothness metric (right) is the spectral arc length and larger values indicate smoother movement [3]. Significance levels are reported: ns (p> 0.05, not represented), $*$ (p< 0.05), $**$ (p< 0.01), $***$ (p< 1.10^{-3}), and $****$ (p< 1.10^{-4}).

6.3.3 Subjective results across participants. The 6 individual NASA TLX scores and the resulting global score (unweighed mean of the 6 individual NASA TLX scores) are averaged to compute subjective metrics across the 12 participants (see Figure 16). The experimental condition was found to have a significant effect on the global score ($F(3, 33) = 18.99, p < 1.10^{-4}$). The participants significantly preferred the conditions **AG** and **CG** over the condition **IG** ($p < 1.10^{-3}$ and $p < 5.10^{-2}$, respectively). They also significantly preferred **AG** over **NG** ($p < 1.10^{-3}$). No other pairwise comparison was found to be statistically significant.

We also found several significant effects of the experimental condition on the individual scores (see Figure 16), namely the mental demand ($F(3, 33) = 4.75, p < 1.10^{-2}$), the perceived performance ($F(3, 33) = 23.974, p < 1.10^{-4}$), the perceived effort ($F(3, 33) = 26.17, p < 1.10^{-4}$), and the frustration ($F(3, 33) = 48.53, p < 1.10^{-4}$). The main pairwise comparison results are as follows:

- the participants found that the **IG** condition resulted in significantly more Effort and Frustration than the conditions **AG** and **CG** ($p < 1.10^{-2}$ for all). Additionally, the **NG** condition resulted in significantly more Effort ($p < 5.10^{-2}$) and Frustration ($p < 1.10^{-2}$) than **AG**;
- the same differences can be noticed in the performance as evaluated subjectively by the participants. Participants found their Performance to be significantly better during condition **CG** compared to both **NG** and **IG** ($p < 1.10^{-2}$). Likewise, participants found their Performance to be significantly better during condition **AG** compared to both **NG** and **IG** ($p < 1.10^{-3}$);
- participants found the condition **AG** to be significantly less mentally demanding than the condition **NG** ($p < 1.10^{-2}$);

The two additional questions that the participants had to answer after each experimental conditions were also analyzed for statistically significant effects (see Figure 17). Participants reported they felt more in control (Q1) under **AG** and **NG** conditions w.r.t. the **IG** condition ($p < 1.10^{-1}$ for both). They also more strongly agreed with the statement Q2 under **AG** and **CG** conditions w.r.t. the **IG** condition ($p < 1.10^{-2}$ for both), indicating that they found the haptic guidance significantly more helpful under **AG** and **CG**.

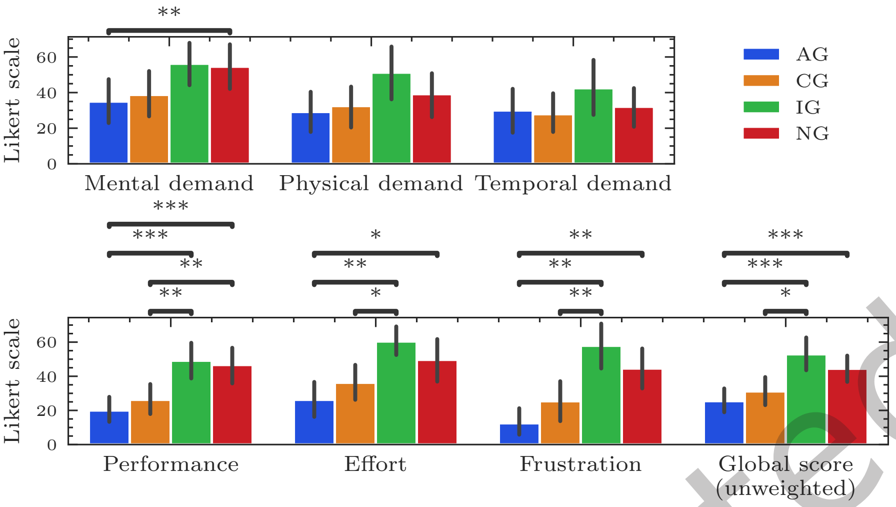

Fig. 16. NASA TLX scores (lower is better). Significance levels are reported: ns (p> 0.05, not represented), $*$ (p< 0.05), $**$ (p< 0.01), $***$ (p< 1.10^{-3}), and $****$ (p< 1.10^{-4}).

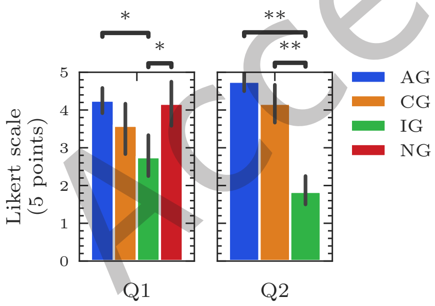

Fig. 17. Scores obtained for the two additional survey questions (see question statements in Table 1).

6.4 Analysis

6.4.1 *Effects of inaccuracies.*

The performance under **CG** condition was found to be significantly better than under condition **IG**. Not only are the execution errors lower and the movements smoother under **CG**, but it was also experienced significantly more positively by participants. They found the **IG** condition to be more demanding, more frustrating, and resulting in a perceived decrease of performance. These quantitative results heavily support the first part of hypothesis **H1**, which states that an inaccurate guidance will increase execution errors and decrease performance.

It was initially hypothesized in **H1** that condition **IG** would lead to performances similar or worse than **NG**. In this study, the level of smoothness was found to be significantly different between **IG** and **NG**: the movements under **IG** were smoother ($p < 0.05$). However, this is the only significant effect we found in objective metrics. As for the subjective metrics, only the answers to Q1 were found to be significantly different ($p < 0.05$), suggesting that participants felt more in control under **NG**. Although not statistically significant, one can also see a trend in the NASA TLX scores (see Figure 16) in favor of **NG** over **IG**.

To conclude, inaccuracies in the models used to generate guidance (i.e., **IG**) significantly degrade performance. Although such inaccurate guidance still improved movement smoothness compared to a non-assisted teleoperation (**NG**), no guidance was overall preferred by the participants. They notably felt more in control without the guidance than with an inaccurate one. However, a correct guidance (**CG**) led to significantly better performances (accuracy, smoothness, and subjective metrics) than both the inaccurate guidance (**IG**) and the absence of guidance (**NG**).

6.4.2 Performance and acceptance of adaptive haptique guidance. Hypothesis **H2.1** was already supported by the results presented in Section 5: correct parameters can be learned from the actions of the operator, even when the initially incorrect parameters are used to guide the operator or when no guidance is generated. The data collected during the study with participants confirmed these results as the mean task model prediction errors were significantly decreased by the learning process (by 90% under **AG** and **NG**). Similarly, the experimental results support **H2.2** that states that **AG** would lead to significantly better performance than **IG**. Compared with the non-adaptive guidance (**IG**), the adaptive guidance scheme (**AG**) significantly improves the performance and was largely preferred by the participants. We did not find a single statistically significant performance difference between **AG** and **CG**, suggesting that the performances under each conditions are similar. The overall trend is that **AG** and **CG** performed similarly, and better than both **NG** and **IG**. Under condition **AG**, movements were found to be more precise and smoother than under both **IG** and **NG**. Furthermore, although participants found the guidance unhelpful under **IG** (see Q2 in Figure 17), they all found the guidance helpful under **AG**. Furthermore, **AG** was ranked as well as **CG** and significantly better than **IG** (and **NG**) in the NASA TLX questionnaire.

6.4.3 Perception of guidance compliance. Although not statistically significant, there is a trend in the subjective results that the condition **AG** is perceived more positively than the condition **CG**. This was not expected, but as it was noticed early on in the study, the participants were asked some questions at the end of their participation beyond the ones in the questionnaires, including whether they preferred **CG** or **AG** and why. Although most had no preference, some of the collected answers showed slight preference for **AG** because it was "more compliant" and some mentioned a phenomenon for **CG** that they described as "damping" along the path.

This "damping" phenomenon is normal and explained by the tracking of ψ_k, whose dynamics tend to smooth the gesture, which is experienced as forces tangent to the planned path by the operator. However, it is present under all condition except **NG**, such that it does not explain the participants feedback. After investigation, it was found that the guidance under the **AG** condition could indeed be more compliant concerning the motion along the path due to a larger covariance, in part due to additional human path deviation and the interactions of the temporal registration with the other parameters. This could explain the fact that **CG** was given a slightly higher level of perceived Frustration and Effort on the NASA TLX questionnaire (see Figure 16). However, note that the difference remains statistically insignificant.

Another explanation is that the guidance path updates, and therefore the evolution of the experienced compliance of the guidance is perceived positively: at startup, under **AG**, the participants can feel that the guidance is incorrect, but then also sense that it is improving over time. Such biases could be further investigated by implementing haptic guidance in such a way that the uncertainty of the parameters does not impact the tracking of ψ and that there is no sensation of damping (similarly to a classical virtual fixture). It should, however, be noted that the damping along the path introduced by the filter improves numerical stability and smooths the guidance, especially when the underlying path is incorrect and in the presence of strong curvature.

7 CONCLUSION AND PERSPECTIVES

7.1 Conclusion

In this article, we have developed a flexible approach to use information provided by an operator *in-the-loop* to augment available information for online task and robot kinematic model parameters learning. The approach is applicable when models of the robot and of the task are known, but that the values of some underlying parameters are not. This is typically the case when a task has been defined but parameters have to be adapted to the actual environment during its execution. Such parameters include, but are not limited to, task and robotic tool registration parameters. An implementation on a real robotic platform with human participants was provided to demonstrate the applicability of such an approach to simultaneously correct a pre-planned trajectory and a nominal robot kinematic model from teleoperation data only. The results further demonstrated that the learning can cope with small variations of the correct parameters, which is of great interest for applications such as robot assisted surgery where, due to environment deformations, the correct task model might change during the execution of the task. As the learning is done online, while the operator is performing the task, the updated parameters can then be used to provide an adaptive assistance that improves over time.

We also presented a comprehensive study with participants that provides results about the effects of both inaccurate and adaptive haptic guidance. In a scenario with initially incorrect task registration and robot tool calibration, the proposed adaptive haptic guidance significantly improved task execution accuracy compared to a non-adaptive haptic guidance. Although the main objective was to evaluate the adaptive guidance approach, the results of this user study are more generally relevant to teleoperation with haptic guidance. From the experimental results, we can conclude that when task and/or robot models are inaccurate, the performance is negatively impacted: the accuracy is reduced, the movements are less smooth, and the user experience is degraded. In this scenario, adaptive guidance was then found to be preferable to non-adaptive guidance, both objectively and subjectively.

7.2 Limitations and perspectives

The proposed fading memory EKF is not the single possible embodiment of the approach and other implementations may be better for different scenarios. For example, moving horizon estimation may perform better when the observations yield very little information locally. Additionally, performing the learning on a window of observations would provide means to implement online numerical tests to make sure that the parameters are identifiable considering the information actually collected. However, storing and exploiting this data would also be much more computationally intensive than the proposed recursive implementation. Something that has not been discussed so far is the detection of modeling errors, as it was assumed that all the parameters were initially incorrect. In practice, the parameters may in some cases be accurate and if they are, it would be better not to update them. Future work will investigate how undesired updates can be avoided, either by detecting modeling errors before starting learning or by learning the parameters without updating the controllers until it is certain that the resulting haptic guidance would improve.

The proposed approach requires a well-structured trajectory, which can for instance be provided by a pre-operative plan in well-codified medical applications. The proposed approach could also be used with less restrictive task models as long as the path is differentiable (e.g., extrapolated trajectory using b-splines). However, in that case, it will not be possible to also update robot parameters without additional sensor data (e.g., full or partial robot pose estimation). The reason is that the learning process exploits the structure of the task to solve the optimization problem, so that without structure, parameter redundancy would appear. Redundancy between robot and task models parameters could also appear in the case of robot models more complex than the pen length used in the presented experiments. As with any parameter estimation method, parameter redundancy would then lead to identifiability issues. By itself this would not prevent the system to guide the user along the correct path. However, the covariance in the Kalman filter might take very large values. In such cases, the model should be re-parameterized, when possible, to reduce the number of parameters.

In the presented experiments, the user benefits from the visualization of the desired path, which is provided through the visual feedback. In real medical applications, the direct feedback may only consist of the tissues as seen from intraoperative images, typically endoscopic images. The preoperative plan, obtained from other diagnostic imaging, could be displayed on a separate screen as an overlay to the medical preoperative images and the user would have to perform the registration mentally, using distinctive features from the scene. Although this work is inspired by challenges from the field of medical robotics, actual implementation on a safety-critical system would require further developments. The main risk is that updating robot parameters could potentially lead to follower robot motion not reflecting the motion of the leader robot. In a safety-critical scenario, the method used by the follower robot to update its parameters from the current estimates should be designed to prevent any undesired follower robot motion. Additionally, it must be acknowledged that the method does not address the correction of dynamic errors, although it was shown to be able to cope with parameters shifts. Therefore, for medical applications, the learning should be conducted during a phase with limited physiological motions.

In complex scenarios, human actions may yield only partial information about the task and robot models' parameters. Nevertheless, this partial information can still be used to infer the parameter values when there are valid hypotheses about the task being performed (e.g., preoperative planning). Using the proposed framework, additional sources of information can be used jointly (e.g., camera, force sensor, distance sensor, etc.) to learn the parameters (see Equation 13). This is out of the scope of this article, but will be investigated in future work. Additionally, the learning process is independent from the haptic guidance and other guidance schemes can be considered. For one, the estimated task model can be used to generate an adaptive Guidance VF by recomputing the guidance reference as the closest point on the path rather than a point-to-point guidance. Also, the learned models can be used to provide visual information through augmented reality. Finally, perspectives include the automatic control of certain DOF of the task (e.g., orientation) using the estimated task and robot models.

A COMPLEMENT TO FIGURE 14

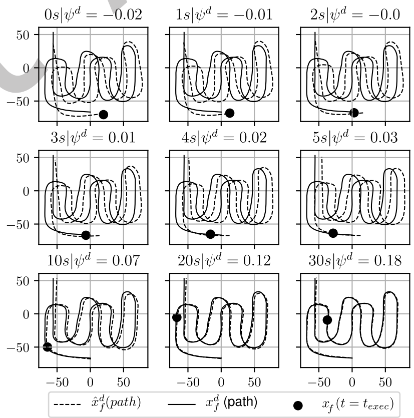

Fig. 18. Correct and estimated trajectories x_f^d displayed in the XY plane of the drawing frame under condition **AG**. For each snapshot, the execution time (not including *clutching* time) and the task progress is provided.

ACKNOWLEDGMENTS

This work was supported by French state funds managed by the ANR within the Investissements d'Avenir programme for the **Labex CAMI** (ANR-11-LABX-0004).

REFERENCES

[1] D. Aarno, S. Ekvall, and D. Kragic. 2005. Adaptive Virtual Fixtures for Machine-Assisted Teleoperation Tasks. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation. 897–903.

[2] Andrea Bajcsy, Dylan P. Losey, Marcia K. O'Malley, and Anca D. Dragan. 2017. Learning robot objectives from physical human interaction. Proceedings of Machine Learning Research 78 (2017), 217–226.

[3] Sivakumar Balasubramanian, Alejandro Melendez-Calderon, Agnes Roby-Brami, and Etienne Burdet. 2015. On the analysis of movement smoothness. 12 (2015).

[4] Lorenzo Bianchi, Francesco Chessa, Andrea Angiolini, Laura Cercenelli, Simone Lodi, Barbara Bortolani, Enrico Molinaroli, Carlo Casablanca, Matteo Droghetti, and Caterina Gaudiano. 2021. The use of augmented reality to guide the intraoperative frozen section during robot-assisted radical prostatectomy. European Urology 80, 4 (2021), 480–488. Publisher: Elsevier.

[5] Bernhard Bischof, Tobias Glück, Martin Böck, and Andreas Kugi. 2018. A path/surface following control approach to generate virtual fixtures. IEEE Transactions on Robotics 34, 6 (2018), 1577–1592. Publisher: IEEE.

[6] Andreea Bobu, Andrea Bajcsy, Jaime F. Fisac, Sampada Deglurkar, and Anca D. Dragan. 2020. Quantifying hypothesis space misspecification in learning from human–robot demonstrations and physical corrections. IEEE Transactions on Robotics 36, 3 (2020), 835–854.

[7] S. A. Bowyer, B. L. Davies, and F. Rodriguez y Baena. 2014. Active Constraints/Virtual Fixtures: A Survey. IEEE Transactions on Robotics 30, 1 (Feb. 2014), 138–157.

[8] Alexander Broad, Ian Abraham, Todd Murphey, and Brenna Argall. 2020. Data-driven Koopman operators for model-based shared control of human–machine systems. The International Journal of Robotics Research 39, 9 (Aug. 2020), 1178–1195. Publisher: SAGE Publications Ltd STM.

[9] Marco Cognetti, Marco Aggravi, Claudio Pacchierotti, Paolo Salaris, and Paolo Robuffo Giordano. 2020. Perception-aware human-assisted navigation of mobile robots on persistent trajectories. IEEE Robotics and Automation Letters 5, 3 (2020), 4711–4718.

[10] J. De Geeter, H. Van Brussel, J. De Schutter, and M. Decreton. 1997. A smoothly constrained Kalman filter. IEEE Transactions on Pattern Analysis and Machine Intelligence 19, 10 (Oct. 1997), 1171–1177.

[11] N. Enayati, E. De Momi, and G. Ferrigno. 2016. Haptics in Robot-Assisted Surgery: Challenges and Benefits. IEEE Reviews in Biomedical Engineering 9 (2016), 49–65.

[12] Jinyu Feng, Xiaojian Li, Xilin Xiao, Bo Ouyang, and Jin Fang. 2021. Virtual Fixtures Assistance for Safe Polyp Dissection in Minimally Invasive Robotic Surgery. In 2021 5th International Conference on Automation, Control and Robots (ICACR). 150–155.

[13] Michael Hagenow, Emmanuel Senft, Robert Radwin, Michael Gleicher, Bilge Mutlu, and Michael Zinn. 2021. Corrective shared autonomy for addressing task variability. IEEE robotics and automation letters 6, 2 (2021), 3720–3727.

[14] Sandra G. Hart and Lowell E. Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In Advances in psychology. Vol. 52. Elsevier, 139–183.

[15] Shervin Javdani, Henny Admoni, Stefania Pellegrinelli, Siddhartha S. Srinivasa, and J. Andrew Bagnell. 2018. Shared autonomy via hindsight optimization for teleoperation and teaming. The International Journal of Robotics Research 37, 7 (June 2018), 717–742.

[16] Theodora Kastritsi, Fotios Dimeas, and Zoe Doulgeri. 2018. Progressive Automation with DMP Synchronization and Variable Stiffness Control. IEEE Robotics and Automation Letters 3, 4 (Oct. 2018), 3789–3796.

[17] T. Liu and M. C. Cavusoglu. 2016. Needle Grasp and Entry Port Selection for Automatic Execution of Suturing Tasks in Robotic Minimally Invasive Surgery. IEEE Transactions on Automation Science and Engineering 13, 2 (April 2016), 552–563.

[18] Dylan P. Losey and Marcia K. O'Malley. 2018. Including Uncertainty when Learning from Human Corrections. In Conference on Robot Learning. 123–132.

[19] Dylan P. Losey and Marcia K. O'Malley. 2019. Learning the Correct Robot Trajectory in Real-Time from Physical Human Interactions. ACM Transactions on Human-Robot Interaction 9, 1 (Dec. 2019), 1–19.

[20] Dylan P. Losey and Marcia K. O'Malley. 2018. Trajectory Deformations From Physical Human–Robot Interaction. IEEE Transactions on Robotics 34, 1 (Feb. 2018), 126–138.

[21] L. Maier-Hein, P. Mountney, A. Bartoli, H. Elhawary, D. Elson, A. Groch, A. Kolb, M. Rodrigues, J. Sorger, S. Speidel, and D. Stoyanov. 2013. Optical techniques for 3D surface reconstruction in computer-assisted laparoscopic surgery. Medical Image Analysis 17, 8 (Dec. 2013), 974–996.

[22] Murilo M. Marinho, Hisashi Ishida, Kanako Harada, Kyoichi Deie, and Mamoru Mitsuishi. 2020. Virtual fixture assistance for suturing in robot-aided pediatric endoscopic surgery. IEEE Robotics and Automation Letters 5, 2 (2020), 524–531. Publisher: IEEE.

[23] Carlo Masone, Paolo Robuffo Giordano, Heinrich H. Bülthoff, and Antonio Franchi. 2014. Semi-autonomous trajectory generation for mobile robots with integral haptic shared control. In 2014 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 6468–6475.

[24] Rocco Moccia, Mario Selvaggio, Luigi Villani, Bruno Siciliano, and Fanny Ficuciello. 2019. Vision-based Virtual Fixtures Generation for Robotic-Assisted Polyp Dissection Procedures. In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE.

[25] Florent Nageotte, Philippe Zanne, Christophe Doignon, and Michel de Mathelin. 2009. Stitching Planning in Laparoscopic Surgery: Towards Robot-assisted Suturing. The International Journal of Robotics Research 28, 10 (Oct. 2009), 1303–1321. Publisher: SAGE Publications Ltd STM.

[26] Alex Tremain Nelson. 2000. Nonlinear estimation and modeling of noisy time series by dual Kalman filtering methods. Ph. D. Dissertation. Oregon Graduate Institute of Science and Technology.

[27] Bojan Nemec, Leon Žlajpah, Sebastjan Šlajpa, Jožica Piškur, and Aleš Ude. 2018. An efficient pbd framework for fast deployment of bi-manual assembly tasks. In 2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids). IEEE, 166–173.

[28] Günter Niemeyer, Carsten Preusche, Stefano Stramigioli, and Dongjun Lee. 2016. Telerobotics. In Springer handbook of robotics. Springer, 1085–1108.

[29] E. Olivieri, G. Barresi, D. G. Caldwell, and L. S. Mattos. 2018. Haptic Feedback for Control and Active Constraints in Contactless Laser Surgery: Concept, Implementation, and Evaluation. IEEE Transactions on Haptics 11, 2 (April 2018), 241–254.

[30] Jeroen van Oosterhout, Jeroen G. W. Wildenbeest, Henri Boessenkool, Cock J. M. Heemskerk, Marco R. de Baar, Frans C. T. van der Helm, and David A. Abbink. 2015. Haptic Shared Control in Tele-Manipulation: Effects of Inaccuracies in Guidance on Task Execution. IEEE Transactions on Haptics 8, 2 (April 2015), 164–175.

[31] Sahba Aghajani Pedram, Changyeob Shin, Peter Walker Ferguson, Ji Ma, Erik P. Dutson, and Jacob Rosen. 2020. Autonomous suturing framework and quantification using a cable-driven surgical robot. IEEE Transactions on Robotics 37, 2 (2020), 404–417. Publisher: IEEE.

[32] Emmanuel Pignat, João Silvério, and Sylvain Calinon. 2020. Learning from demonstration using products of experts: Applications to manipulation and task prioritization. The International Journal of Robotics Research (2020), 02783649211040561. Publisher: SAGE Publications Sage UK: London, England.

[33] Thibault Poignonec, Florent Nageotte, and Bernard Bayle. 2023. A fading memory discontinuous EKF for the online model identification of cable-driven robots with backlash. In 2023 American Control Conference (ACC). IEEE, 949–954.

[34] Thibault Poignonec, Florent Nageotte, Nabil Zemiti, and Bernard Bayle. 2021. Simultaneous haptic guidance and learning of task parameters during robotic teleoperation – a geometrical approach. In 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 3619–3625.

[35] Thibault Poignonec, Philippe Zanne, Benoît Rosa, and Florent Nageotte. 2020. Towards In Situ Backlash Estimation of Continuum Robots Using an Endoscopic Camera. IEEE Robotics and Automation Letters 5, 3 (July 2020), 4788–4795.

[36] Gennaro Raiola, Susana Sanchez Restrepo, Pauline Chevalier, Pedro Rodriguez-Ayerbe, Xavier Lamy, Sami Tliba, and Freek Stulp. 2018. Co-manipulation with a library of virtual guiding fixtures. Autonomous Robots 42, 5 (2018), 1037–1051.

[37] James B. Rawlings. 2015. Moving Horizon Estimation. In Encyclopedia of Systems and Control, John Baillieul and Tariq Samad (Eds.). Springer, London, 799–804.

[38] Rob Reilink, Stefano Stramigioli, and Sarthak Misra. 2013. Image-based hysteresis reduction for the control of flexible endoscopic instruments. Mechatronics 23, 6 (Sept. 2013), 652–658.

[39] Jing Ren, Rajni V. Patel, Kenneth A. McIsaac, Gerard Guiraudon, and Terry M. Peters. 2008. Dynamic 3-D Virtual Fixtures for Minimally Invasive Beating Heart Procedures. IEEE Transactions on Medical Imaging 27, 8 (Aug. 2008), 1061–1070.

[40] Susana Sánchez Restrepo, Gennaro Raiola, Pauline Chevalier, Xavier Lamy, and Daniel Sidobre. 2017. Iterative virtual guides programming for human-robot comanipulation. In 2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM). IEEE, 219–226.

[41] Leonel Rozo. 2019. Interactive trajectory adaptation through force-guided bayesian optimization. In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 7596–7603.

[42] Golnoosh Samei, Keith Tsang, Claudia Kesch, Julio Lobo, Soheil Hor, Omid Mohareri, Silvia Chang, S. Larry Goldenberg, Peter C. Black, and Septimiu Salcudean. 2020. A partial augmented reality system with live ultrasound and registered preoperative MRI for guiding robot-assisted radical prostatectomy. Medical image analysis 60 (2020), 101588. Publisher: Elsevier.

[43] Mario Selvaggio, Fei Chen, Boyang Gao, Gennaro Notomista, Francesco Trapani, and Darwin Caldwell. 2016. Vision based virtual fixture generation for teleoperated robotic manipulation. In 2016 International Conference on Advanced Robotics and Mechatronics (ICARM). 190–195. ISSN: null.

[44] M. Selvaggio, G. A. Fontanelli, F. Ficuciello, L. Villani, and B. Siciliano. 2018. Passive Virtual Fixtures Adaptation in Minimally Invasive Robotic Surgery. IEEE Robotics and Automation Letters 3, 4 (Oct. 2018), 3129–3136.

[45] Dan Simon. 2006. Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches. John Wiley & Sons, Inc.

[46] Dan Simon. 2010. Kalman Filtering with State Constraints: A Survey of Linear and Nonlinear Algorithms. Control Theory & Applications, IET 4 (Sept. 2010), 1303–1318.

[47] J. Smisek, M. M. van Paassen, and A. Schiele. 2015. Haptic guidance in bilateral teleoperation: Effects of guidance inaccuracy. In 2015 IEEE World Haptics Conference (WHC). 500–505.

[48] Gaelle Thomas, Laurent Barbé, Benoît Larrat, Pauline Agou, Jonathan Vappou, and Florent Nageotte. 2020. Planning Framework for Robot-assisted Blood-Brain Barrier Opening with Focused Ultrasound. In 2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC). IEEE, 5033–5036.

[49] Jocelyne Troccaz. 2013. Medical robotics. John Wiley & Sons.

[50] Jozef Vörös. 2010. Modeling and identification of systems with backlash. Automatica 46, 2 (2010), 369–374.

[51] Long Wang, Zihan Chen, Preetham Chalasani, Rashid M. Yasin, Peter Kazanzides, Russell H. Taylor, and Nabil Simaan. 2017. Force-controlled exploration for updating virtual fixture geometry in model-mediated telemanipulation. Journal of Mechanisms and Robotics 9, 2 (2017). Publisher: American Society of Mechanical Engineers Digital Collection.

[52] Lin Yang, Yanan Li, Deqing Huang, Jingkang Xia, and Xiaodong Zhou. 2022. Spatial Iterative Learning Control for Robotic Path Learning. IEEE Transactions on Cybernetics (2022), 1–10.

[53] Rashid Yasin, Preetham Chalasani, Nicolas Zevallos, Mahya Shahbazi, Zhaoshuo Li, Anton Deguet, Peter Kazanzides, Howie Choset, Russell H. Taylor, and Nabil Simaan. 2021. Evaluation of Hybrid Control and Palpation Assistance for Situational Awareness in Telemanipulated Task Execution. IEEE Transactions on Medical Robotics and Bionics 3, 1 (Feb. 2021).

[54] Mohammad Kassem Zein, Abbas Sidaoui, Daniel Asmar, and Imad H. Elhajj. 2020. Enhanced Teleoperation Using Autocomplete. In 2020 IEEE International Conference on Robotics and Automation (ICRA). 9178–9184.

[55] Lucile Zorn, Pierre Renaud, Bernard Bayle, Laurent Goffin, Cyrille Lebosse, Michel de Mathelin, and Jack Foucher. 2011. Design and evaluation of a robotic system for transcranial magnetic stimulation. IEEE Transactions on Biomedical Engineering 59, 3 (2011), 805–815. Publisher: IEEE.

Received 18 March 2024; revised 25 August 2024; accepted 10 October 2024