Co-manipulation with a Library of Virtual Guiding Fixtures

Gennaro Raiola, Susana Sanchez Restrepo, Pauline Chevalier, Pedro Rodriguez-Ayerbe, Xavier Lamy, Sami Tliba, Freek Stulp

Noname manuscript No.
(will be inserted by the editor)

Co-manipulation with a Library of Virtual Guiding Fixtures

Gennaro Raiola · Susana Sanchez Restrepo · Pauline Chevalier · Pedro Rodriguez-Ayerbe · Xavier Lamy · Sami Tliba · Freek Stulp

the date of receipt and acceptance should be inserted later

Abstract Virtual guiding fixtures constrain the movements of a robot to task-relevant trajectories, and have been successfully applied to, for instance, surgical and manufacturing tasks. Whereas previous work has considered guiding fixtures for single tasks, in this paper we propose a library of guiding fixtures for multiple tasks, and propose methods for 1) Creating and adding guides based on machine learning; 2) Selecting guides on-line based on probabilistic implementation of guiding fixtures; 3) Refining existing guides based on an incremental learning method. We demonstrate in an industrial task that a library of guiding fixtures provides an intuitive haptic interface for joint human-robot completion of tasks, and improves performance in terms of task execution time, mental workload and errors.

G. Raiola[1,2,4]
E-mail: gennaro.raiola@ensta-paristech.fr

S. Sanchez Restrepo[2]
E-mail: susana.sanchezrestrepo@cea.fr

P. Chevalier[1]
E-mail: pauline.chevalier@ensta-paristech.fr

P. Rodriguez-Ayerbe[3]
E-mail: pedro.rodriguez@centralesupelec.fr

X. Lamy[2]
E-mail: xavier.lamy@cea.fr

S. Tliba[3]
E-mail: sami.tliba@lss.supelec.fr

F. Stulp[1,4,5]
E-mail: freek.stulp@dlr.de

[1] Robotics and Computer Vision, ENSTA-ParisTech, Palaiseau, France · [2] CEA-List, Gif-sur-Yvette, France · [3] Univ. Paris-Sud, CNRS, CentraleSupelec, Gif-sur-Yvette, France · [4] FLOWERS Team, INRIA, Bordeaux Sud-Ouest, France · [5] German Aerospace Center (DLR), Institute of Robotics and Mechatronics, Wessling, Germany

Keywords Human robot collaborative tasks in manufacturing · Learning from demonstration · Virtual fixture

1 Introduction

Recent improvements in the safety and (force) sensing capabilities of robots now enable humans to physically interact and solve tasks collaboratively with robots. The advantages of this collaboration is that it enables non-expert users to quickly teach robots new behaviours for new tasks. This is essential for modern assembly lines, where lot sizes are becoming ever smaller due to customization, and high degrees of flexibility are necessary to quickly adapt to changing markets (Hermann et al, 2016).

This flexibility and teach-in programming requires robots to be adaptive and to predict the intentions of humans, for which machine learning is a key enabler. In this paper, we apply machine learning and probabilistic methods to "virtual guiding fixtures" (Lin et al, 2006), so that non-expert users can teach new fixtures, and the robot is able to recognize on-line which fixture the human intends to select.

A virtual guiding fixture (Lin et al, 2006) constrains the motion of an end-effector to certain task-relevant trajectories. A well-known example of a guiding fixture from everyday life is the ruler, which allows us to draw very straight lines by constraining the movement of the pen tip along a 1-D trajectory on the 2-D paper. Robots are able to implement more complex virtual guiding fixtures, as illustrated in Fig. 1.

Whereas previous work has focussed on single virtual guides for single tasks, here we consider scenarios in which multiple tasks must be solved, and a library of virtual guides is thus necessary. Therefore, to maintain and evaluate such a library, we make the following contributions[1]:

- Apply incremental training of Gaussian Mixture Models (GMM), as previously used for Programming by Demonstration (PbD) in (Calinon, 2007), to create and refine virtual guides (Section 4 and 6).
- Define a controller to select among multiple virtual guides based on a probabilistic implementation of them (Section 5), which constitutes the main technical contribution of our work.
- Present an user study (Section 7), in which we evaluate the usability and impact of the library of virtual guides in the context of an industrial pick-and-place task.

The rest of this paper is structured as follows. In the next section, we discuss related work. In Section 3, we introduce

[1] Our previous work (Raiola et al, 2015a,b) focussed on the theoretical framework underlying multiple virtual guides, as well as an analysis of their stability. This paper focusses instead on the pragmatic implementation of a library of such guides, including a full user study.

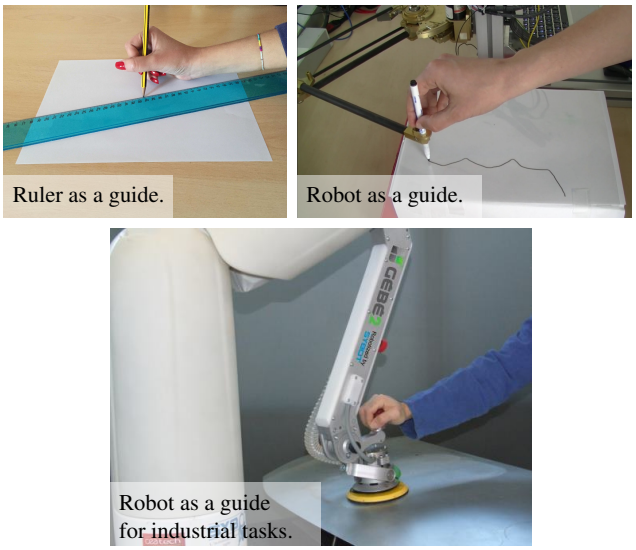

Fig. 1: Rulers simplify the drawing of lines, because they constrain (guide) the movement of the pencil tip (top left). Robots can similarly constrain human motions with virtual guides, but allow more flexibility on the shape of the guide (top right). Such virtual guides enable co-manipulation for industrial tasks (bottom).

a possible way to generate virtual guides by using virtual mechanisms as proposed in (Joly and Andriot, 1995), which forms the background of our work. In Section 4, we introduce how to create and add guides to the library. In Section 5 we present the controller which enables the on-line selection of multiple virtual guides. In Section 6 we discuss how to refine existing guides based on the incremental training of GMM. We present the user study in Section 7, and conclude with Section 8.

2 Related Work

Virtual guides are used to enforce virtual constraints on the movements of robots, in order to assist the user during a collaborative task. Virtual fixtures are especially useful in contexts where human decision making is still required to perform the overall task, but where constraints on the accuracy or required forces of the motion preclude humans from performing such tasks without robot assistance. Virtual fixtures were first introduced by (Rosenberg, 1993), where virtual fixtures are presented as an overlay of augmented sensory information on a workspace used to improve human performance in a teleoperated manipulation task. The fundamental concept is that virtual fixtures can reduce mental workload, task time, and errors during the collaborative task. After Rosenberg's initial work, the use of virtual fixtures has been extended to robotic surgery under the name of *active constraints* (Ho et al, 1995; Davies et al, 2006) and to industrial applications by (Colgate et al, 2003) in the context of *Intelligent Assist Devices*.

Nowadays, virtual fixtures has been featured in several different works, but unfortunately "there is currently no definitive concept which unifies the field" (Bowyer et al, 2014) because of the different definitions, applications and implementation methods. Generally, virtual fixtures has been used in teleoperation or comanipulation contexts. In teleoperation, the user controls a slave robot via a separate master device (Joly and Andriot, 1995; Aarno et al, 2005; Abbott, 2005; Bowyer and y Baena, 2013), this offers benefits such as motion scaling and the possibility to operate in restricted and unsafe environments, for example (Ryden et al, 2013) use virtual guides to teleoperate an underwater robot, while (David et al, 2014) proposed a supervisory control system to speed up a disk-cutter insertion process.

In a comanipulation context, the user directly interacts with the robot through physical contact (Raiola et al, 2015a; Becker et al, 2013; Dumora, 2014; Pezzementi et al, 2007). This allows a direct interaction between the robot and the user, and a more intuitive ability to perform the task since the user is better integrated in the procedure compared to the case where the user interacts with the environment through a teleoperated robot. The type of assistance offered by the virtual fixtures can vary among different definitions, but in general they are either used to guide the user along a task-specific pathway or to limit the user to move the robot within a safe region.

The particular implementation of virtual guides we use is based on the work presented by (Joly and Andriot, 1995), where a passive virtual mechanism is connected to the robot end-effector by a spring-damper system in a teleoperation context. Instead, we use the virtual mechanisms in a co-manipulation framework, i.e. the user is directly in contact with the robot. Virtual mechanisms have also been used by (Pezzementi et al, 2007), where they are called "proxies". Virtual guides may also be implemented by using anisotropic admittances to attenuate the non-preferred user force component (Marayong et al, 2003; Bettini et al, 2004). These methods require sensing external inputs, such as the force or the velocity applied by the user on the robot end-effector. This is not required with our control scheme.

Regarding the way virtual fixtures can be created, there are many possible solutions since there are different possible applications where they can be useful, usually the way to create them is strictly related to the goals of the application. In general, virtual fixtures have often been limited to predefined geometric shapes (Marayong et al, 2003) or combinations of shapes (Aarno et al, 2005; Kuang et al, 2004) or defined through well-defined geometric models (Joly and Andriot, 1995; Dumora, 2014). On the other hand, programming by demonstration (PbD) appears as a promising solution to program robots in a fast and simple way when the

task is known by the user. In PbD, teaching a path usually involves demonstrating the set of trajectories and retrieving a generalized representation of the data set suitable for reproduction by a robot. Generating guides from demonstrations has been explored by (Aarno et al, 2005), who model demonstrations in a segmented sequence of straight lines. Another interesting work about virtual fixtures and programming by demonstrations has been conducted by (Yoon et al, 2014). In this work the authors *personalize* the virtual fixture based on a set of demonstrations provided by the users in order to match their preferences about the guidance.

In our work, we use the demonstrations of the user to train Gaussian Mixture Models (GMM) as in (Calinon et al, 2007), which ensures smooth movements and explicitly models the variance in user demonstrations. Moreover, this allows us to define one of the novel aspect of our work, i.e. the *probabilistic* virtual guides. A first advantage of the probabilistic approach is that it enables a guide to be activated/deactivated based on the probability of belonging to it, which leads to smooth transitions. This is preferable to switching the guide on/off as in (Li and Okamura, 2003; Aarno et al, 2005; Yu et al, 2005), and does not require the manual design of distance thresholds for activation, as in (Nolin et al, 2003).

A second advantage is that the probabilistic approach allows us to simultaneously activate and recognize several guides, by assigning probabilities to each guide based on user behavior. Thus, our method enables the use of a *library of guides*, with one guide for each distinct task. Multiple guides have been previously used, but these (sub)guides are activated sequentially for one unique task, rather than in parallel for several tasks. For instance (Kuang et al, 2004) combine different shape primitives to facilitate maze navigation. (Aarno et al, 2005) use HMM to probabilistically choose a guide in a sequence of linear guides to accomplish a pick and place task.

Finally, we consider a co-manipulation instead of teleoperation framework, as is customary with virtual guides. In this respect, our work can be compared to (Amor et al, 2014; Medina et al, 2012; Rozo et al, 2016; Wrede et al, 2013) where the user and the robot have to execute a learned task together. Regarding the definition of the virtual guides through PbD, our work can be compared to the work done by (Vakanski et al, 2012; Mollard et al, 2015; Boy et al, 2007; Ewerton et al, 2016; Lee and Ott, 2011; Sanchez Restrepo et al, 2017) where the concept of *Task refinement* is exploited, as we will see in Section 6 this is possible due to the incremental training of GMM (Calinon, 2007).

3 Background: Virtual Mechanisms as Virtual Guides

We implement a virtual guide as a connection between the end-effector of the robot and a simulated virtual robot called "virtual mechanism" (Joly and Andriot, 1995). The equations and control schemes in (Joly and Andriot, 1995) are important background knowledge to understand our contributions, so we provide an overview of them in this section.

In general the virtual robot has fewer degrees of freedom than the real one, and thus the movements of the real robot are constrained by the possible movements of the virtual robot, see Fig. 2.

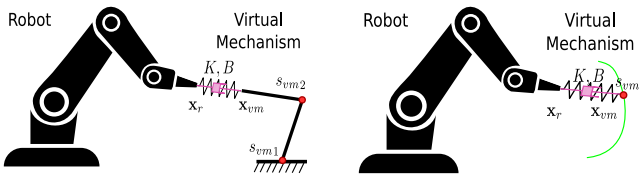

Fig. 2: Left: A virtual mechanism is a virtual (spring-damper) connection between the robot and the virtual robot with fewer degrees of freedom. $\mathbf{x}_{\text{vm}}$ and $\mathbf{x}_{\text{r}}$ represent respectively the end-effector position in Cartesian space of the virtual mechanism and the robot. s_{vmi} represents the degree of freedom of the virtual mechanism. Right: In our work, the virtual mechanism has only one degree of freedom represented by s_{vm}, which represents the movement along a trajectory, the virtual guide. The virtual mechanism can be thought as a cart moving along a rail, with the rail acting as the constraint.

The robot end-effector and the virtual "cart" mechanism are coupled by a spring-damper system. In this way if the robot end-effector moves, the cart is pulled along the rail in the direction of the movement, on the other hand, the cart also pulls the robot towards the rail, because the connection pulls in both directions. The overall effect is that the robot end-effector can be moved easily along the virtual rail, but not away from the rail. The position of the cart on the rail in Cartesian space is described by $\mathbf{x}_{\text{vm}}$. The distance it has traveled along the rail is function of the phase s_{vm}, with $s_{\text{vm}} = 0$ at the beginning and $s_{\text{vm}} = 1$ at the end of the rail, as illustrated in Fig. 3. The kinematics of the virtual mechanism is described by:

$$\mathbf{x}_{\text{vm}} = f(s_{\text{vm}}), \tag{1}$$

$$\dot{\mathbf{x}}_{\text{vm}} = \mathbf{J}_{\text{vm}}(s_{\text{vm}})\dot{s}_{\text{vm}}. \tag{2}$$

In Section 5.1 we will describe how to implement the functions $f(s_{\text{vm}})$ and $\mathbf{J}_{\text{vm}}(s_{\text{vm}})$ from user demonstrations.

3.1 Force on the virtual mechanism

The virtual mechanism is connected to the robot end-effector with a virtual spring-damper system. The force ap-

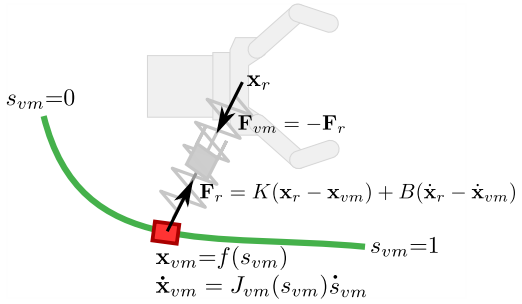

Fig. 3: The main variables and equations of the virtual mechanism.

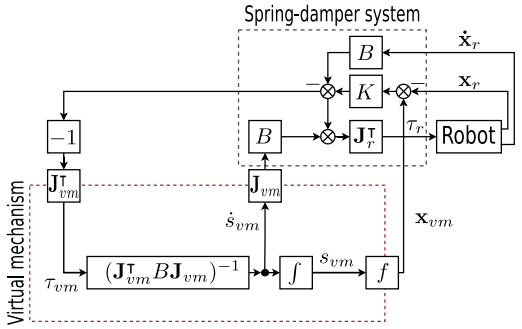

Fig. 4: Control scheme for the virtual mechanism.

plied to the virtual mechanism by the robot is:

$$\mathbf{F}_\mathrm{r} = K(\mathbf{x}_\mathrm{r} - \mathbf{x}_\mathrm{vm}) + B(\dot{\mathbf{x}}_\mathrm{r} - \dot{\mathbf{x}}_\mathrm{vm}). \tag{3}$$

The virtual mechanism is ideal, so the efforts applied on it are null

$$\mathbf{J}_\mathrm{vm}{}^\intercal \mathbf{F}_\mathrm{r} = 0, \tag{4}$$

which leads to

$$\mathbf{J}_\mathrm{vm}{}^\intercal (K(\mathbf{x}_\mathrm{r} - \mathbf{x}_\mathrm{vm}) + B(\dot{\mathbf{x}}_\mathrm{r} - \mathbf{J}_\mathrm{vm}\dot{s}_\mathrm{vm})) = 0. \tag{5}$$

By solving (5) with respect to $\dot{s}_\mathrm{vm}$, we obtain a first order dynamical system that expresses the evolution of the virtual cart along the virtual rail:

$$\dot{s}_\mathrm{vm} = (\mathbf{J}_\mathrm{vm}{}^\intercal B \mathbf{J}_\mathrm{vm})^{-1} \mathbf{J}_\mathrm{vm}{}^\intercal (K(\mathbf{x}_\mathrm{r} - \mathbf{x}_\mathrm{vm}) + B\dot{\mathbf{x}}_\mathrm{r}). \tag{6}$$

Moving the robot end-effector away from the virtual cart ($\mathbf{x}_\mathrm{r} \neq \mathbf{x}_\mathrm{vm}$) will thus make it slide along the rail, with a velocity described by (6)[2].

[2] Eq. (6) contains the inverse of the matrix $(\mathbf{J}_\mathrm{vm}{}^\intercal B \mathbf{J}_\mathrm{vm})$, which may lead to singularities. This problem and possible solutions are presented in Section 2.3 of (Raiola, 2017).

3.2 Force on the robot end-effector

Because the virtual mechanism and the robot end-effector are connected to *each other*, the virtual mechanism also applies a force on the robot end-effector, i.e.

$$\mathbf{F}_\mathrm{vm} = -\mathbf{F}_\mathrm{r} = K(\mathbf{x}_\mathrm{vm} - \mathbf{x}_\mathrm{r}) + B(\dot{\mathbf{x}}_\mathrm{vm} - \dot{\mathbf{x}}_\mathrm{r}). \tag{7}$$

This virtual force can be transformed into actual control commands for the robot, for instance with a compliance controller. In our implementation, we used the robot's Jacobian transposed $\mathbf{J}_\mathrm{r}^\intercal$ to convert the forces into torque references for the motor controllers. Fig. 4 illustrates the signals connections between the robot and the virtual mechanism.

4 Creating and Adding Guides

In the previous section, we explained how a virtual guide is implemented as a virtual mechanism. In this paper, the mechanism may be considered as a virtual cart on a rail (a 3D trajectory), which is connected to the robot end-effector with a spring-damper system. In this section, we present a method for adding a new guide (rail) to a library of guides through demonstrations and machine learning.

4.1 Gaussian Mixture Model

In our approach, virtual guides are extracted from (multiple) user demonstrations by training a Gaussian Mixture Model with Expectation Maximization, as in (Calinon et al, 2007).

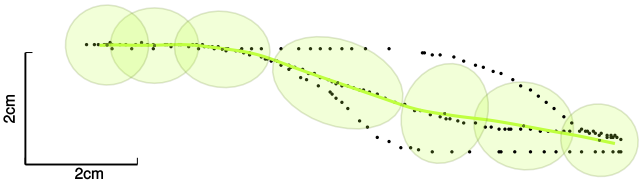

Fig. 5: Example of a Gaussian Mixture Model, trained on three demonstrated trajectories

In a GMM, the demonstrated data is modelled by a mixture of K components defined by a probability density function:

$$p(\zeta_m) = \sum_{k=1}^{K} p(k)p(\zeta_m|k), \tag{8}$$

where $\{\zeta_m\}_{m=1}^{M}$ represents the demonstrated set of Cartesian points of dimension D, $p(k)$ is the prior probability and $p(\zeta_m|k)$ is the conditional probability. A Gaussian Mixture Model can be fully described by its parameters θ which are

$\theta = \{\pi_k, \mu_k, \Sigma_k, M\}_{k=1}^{K}$, respectively the priors, the means, the covariance matrices and the number of samples[3] in ζ.

For a mixture of K components of dimensionality D the parameters in (8) are defined as:

$$\begin{aligned} p(k) &= \pi_k, \\ p(\zeta_m|k) &= \mathcal{N}(\zeta_m; \mu_k, \Sigma_k), \qquad (9) \\ &= \frac{e^{\left(-\frac{1}{2}(\zeta_m-\mu_k)^\intercal {\Sigma_k}^{-1}(\zeta_m-\mu_k)\right)}}{\sqrt{(2\pi)^D|\Sigma_k|}}. \qquad (10) \end{aligned}$$

The log-likelihood of the model described by θ, given a set of M datapoints $\{\zeta_m\}_{m=1}^{M}$ is:

$$\mathcal{L}(\theta) = \frac{1}{M}\sum_{m=1}^{M} \ln(p(\zeta_m)). \qquad (11)$$

Where $p(\zeta_m)$ is the probability that ζ_m has been generated by the model, which is computed using (8).

4.1.1 Training the Gaussian Mixture Model

The GMM is initially trained from user demonstrations, using the approach described in (Calinon et al, 2007; Raiola et al, 2015a), and briefly repeated here. The demonstrated trajectories consist of samples $\{\zeta_m = [x(t_m), y(t_m), z(t_m)]\}_{m=1}^{M}$. Multiple trajectories are first aligned using Dynamic Time Warping. Then, each sample in a trajectory is associated with a phase value defined as $s(t_m) = (t_m - t_1)/(t_M - t_1)$, i.e. $s(t_1) = 0$ at the beginning of the demonstration, and $s(t_M) = 1$ at the end. The resulting samples in one trajectory then have the format $\{[x_m, y_m, z_m, s_m]\}_{m=1}^{M}$.

Fitting the GMM to this data is done with the Expectation-Maximization algorithm (EM). EM incrementally adjusts the priors π_k and the parameters μ_k and Σ_k of the Gaussian functions to fit the data until a stop criterion is met. This algorithm guarantees monotone increase of the likelihood of the training set during optimization.

4.2 Gaussian Mixture Regression

Virtual mechanisms require implementations of the kinematics equations $\mathbf{x}_{\text{vm}} = f(s_{\text{vm}})$ (1) and $\dot{\mathbf{x}}_{\text{vm}} = \mathbf{J}_{\text{vm}}(s_{\text{vm}})\dot{s}_{\text{vm}}$ (2). These are extracted from the GMM through Gaussian Mixture Regression (GMR). In the context of a virtual mechanism, the input space is S, and the output space is X, respectively the phase s_{vm} and virtual mechanism's position $\mathbf{x}_{\text{vm}}$. Given this partition, the mean and covariance matrix[4] are decomposed as

$$\mu_k = [\mu_{k,S}^\intercal, \mu_{k,X}^\intercal]^\intercal \text{ and } \Sigma_k = \begin{bmatrix} \Sigma_{k,S} & \Sigma_{k,SX} \\ \Sigma_{k,XS} & \Sigma_{k,X} \end{bmatrix}, \qquad (12)$$

The implementation of $\mathbf{x}_{\text{vm}} = f(s_{\text{vm}})$ in (1) corresponds to computing $\overline{\mathbf{x}}_{\text{vm}} = E(\mathbf{x}_{\text{vm}}|s_{\text{vm}})$, i.e. the expectation of $\mathbf{x}_{\text{vm}}$ given the input s_{vm}:

$$\overline{\mathbf{x}}_{\text{vm}} = \sum_{k=1}^{K} \beta_k(s_{\text{vm}})(\mu_{k,X} + \Sigma_{k,XS}\Sigma_{k,S}^{-1}(s_{\text{vm}} - \mu_{k,S})), \qquad (13)$$

with:

$$\beta_k(s_{\text{vm}}) = \frac{\pi_k g(\mathbf{x}; \mu_{k,S}, \Sigma_{k,S})}{\sum_{l=1}^{K} \pi_l g(\mathbf{x}; \mu_{l,S}, \Sigma_{l,S})} = \frac{\pi_k g(\mathbf{x}; s_{\text{vm}}^k)}{\sum_{l=1}^{K} \pi_l g(\mathbf{x}; s_{\text{vm}}^l)}. \qquad (14)$$

The function g represents a Gaussian distribution defined as:

$$g(\mathbf{x}; \mu, \Sigma) = \frac{e^{\left(-\frac{1}{2}(\mathbf{x}-\mu)^\intercal \Sigma^{-1}(\mathbf{x}-\mu)\right)}}{\sqrt{(2\pi)^D|\Sigma|}}. \qquad (15)$$

The function $\mathbf{J}_{\text{vm}}(s_{\text{vm}})$ in (2) is implemented with the analytical derivative of (13) in respect of s_{vm}.

In summary, the kinematics of the virtual mechanism $\mathbf{x}_{\text{vm}} = f(s_{\text{vm}})$ is computed with Gaussian Mixture Regression (13), based on a Gaussian Mixture Model (8), whose parameters are trained by applying the Expectation Maximization algorithm to a set of demonstrated trajectories.

5 Selecting Guides

In a library of virtual guides, different guides exist to solve different tasks, see Fig. 6. Our aim is to enable the robot to recognize on-line which task the user intends to solve. To avoid abrupt switches, we implement a control scheme in which all mechanisms are simultaneously active, but scaled with the probability that the task with which the mechanism is associated is being solved. Thus, the final force $\mathbf{F}_{\text{res}}$ applied to the end-effector is a weighted sum of the forces from each guide $\mathbf{F}_{\text{vm}}^{n=1\ldots N}$:

$$\mathbf{F}_{\text{res}} = \sum_{n=1}^{N} p_n \mathbf{F}_{\text{vm}}^n. \qquad (16)$$

Our approach requires the computation of the probabilities $p_{n=1\ldots N}$, which represent the probability that the n^{th} guide is responsible for the current task. To do so, we first propose "probabilistic virtual mechanisms", show how they enable $p_{n=1\ldots N}$ to be computed, and propose different interaction modes based on the exact scaling in (16).

[3] Note that M is not strictly necessary to describe the model but it will be useful for the incremental training.

[4] The covariance matrix $\Sigma_{e,S}$ is actually a scalar, because the phase is always 1-dimensional. For consistency, we nevertheless use the bold symbol Σ rather than σ^2.

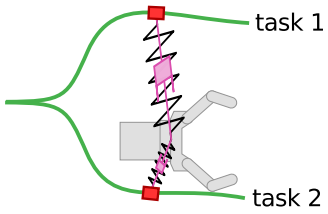

Fig. 6: Multiple virtual mechanisms – one for each task – simultaneously connected to the robot end-effector.

5.1 Probabilistic Virtual Mechanism

We define a probabilistic virtual mechanism as a virtual mechanism in which there is uncertainty about the position of the virtual end-effector, i.e. the position of the cart on the rail. This uncertainty is represented by a Gaussian distribution, as visualized in Fig. 7.

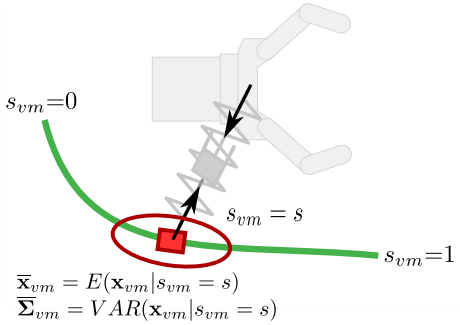

Fig. 7: The current state of the virtual mechanism is modelled as a multi-variate Gaussian distribution $\mathcal{N}(\overline{\mathbf{x}}_{\text{vm}}, \Sigma_{\text{vm}})$.

The covariance matrix of the distribution is readily computed from the Gaussian Mixture Model through Gaussian Mixture Regression by computing the conditional variance $VAR(\mathbf{x}_{\text{vm}}|s_{\text{vm}})$:

$$\Sigma_{\text{vm}} = \sum_{k=1}^{K} \beta_k(s_{\text{vm}})^2 \left(\Sigma_{k,X} - \Sigma_{k,XS}\Sigma_{k,S}^{-1}\Sigma_{k,XS}^{\mathsf{T}}\right). \tag{17}$$

Thus, the (uncertain) position of the virtual mechanism is represented by the Gaussian distribution:

$$\overline{\mathbf{x}}_{\text{vm}} = E(\mathbf{x}_{\text{vm}}|s_{\text{vm}} = s), \tag{18}$$

$$\Sigma_{\text{vm}} = VAR(\mathbf{x}_{\text{vm}}|s_{\text{vm}} = s). \tag{19}$$

We now show how the probabilistic virtual mechanism is used to compute the probabilities $p_{n=1\ldots N}$ for each of the N guides in the library.

5.2 Probabilistic Weighting

Fig. 8 illustrates the association problem when using multiple guides. The two Gaussian Mixture Models represent the two virtual guides, which are associated with two different tasks. The inset to the right shows how the robot end-effector $\mathbf{x}_r$ is connected to both of the virtual mechanism (the "carts"). Because $\mathbf{x}_r$ is closer to the lower cart 2, it is more likely that the user intends to execute task 2, and the force exerted by the cart 2 should be higher than that exerted by cart 1. This intuition is implemented with the probabilistic weighting scheme.

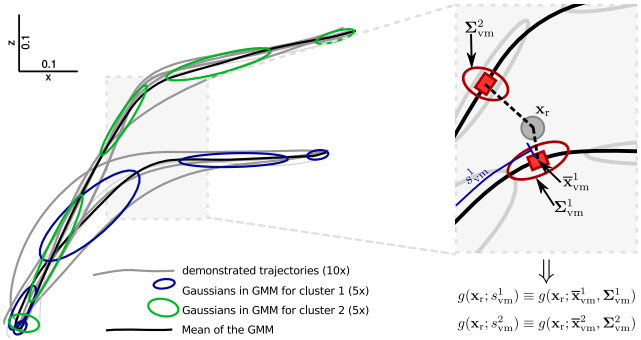

Fig. 8: Left: demonstrated trajectories (light gray) and the two GMMs. Right: Relevant variables for computing $g(\mathbf{x}_{\text{r}}; s_{\text{vm}})$.

If we have N virtual mechanisms, there are N cart positions $\mathbf{x}_{\text{vm}}^{n=1:N}$, and N probabilities. The probability p_n that the n^{th} cart is responsible for guiding the end-effector at position $\mathbf{x}_{\text{r}}$ is

$$p(n; \mathbf{x}_{\text{r}}, s_{\text{vm}}^n) = \frac{g(\mathbf{x}_{\text{r}}; \mu_{\text{vm}}^n, \Sigma_{\text{vm}}^n)}{\sum_{i=1}^{N} g(\mathbf{x}_{\text{r}}; \mu_{\text{vm}}^i, \Sigma_{\text{vm}}^i)} = \frac{g(\mathbf{x}_{\text{r}}; s_{\text{vm}}^n)}{\sum_{i=1}^{N} g(\mathbf{x}_{\text{r}}; s_{\text{vm}}^i)}, \tag{20}$$

where the means and covariance matrices of the cart position are determined from the cart phase s_{vm} with (13) and (17) respectively.

Each of the N virtual mechanisms applies a force $\mathbf{F}_{\text{vm}}^n$ to the end-effector. The relative influence of each VM is scaled with the probability $p(n; \mathbf{x}_{\text{r}}, s_{\text{vm}}^n)$, so that the resultant force on the end-effector is[5]:

$$\mathbf{F}_{\text{res}} = \sum_{n=1}^{N} p(n; \mathbf{x}_{\text{r}}, s_{\text{vm}}^n)\mathbf{F}_{\text{vm}}^n. \tag{21}$$

As described in (Raiola et al, 2015a), the underlying assumption in using (21) is that $\mathbf{x}_{\text{r}}$ *must* belong to one of the VMs[6]. Another approach is to assume that if $\mathbf{x}_{\text{r}}$ is too far from the VMs, it does not belong to any of the VMs. To do so, we use a Gaussian function $h(\mathbf{x}_{\text{r}}; \mu_{\text{vm}}, \Sigma_{\text{vm}})$ combined with (21), i.e. a probability density function as in (10), but

[5] The stability of such probabilistically weighted virtual guides is analyzed in (Raiola et al, 2015b).

[6] For this reason, we call the resulting guides "Hard Guides"

without the normalization factor $\sqrt{(2\pi)^k|\Sigma_{\text{vm}}|}$, as the Gaussian function has a known maximum of 1.

$$h(\mathbf{x},\mathbf{x}_{\text{vm}}) = e^{\left(-\frac{1}{2}(\mathbf{x}-\mathbf{x}_{\text{vm}})^{\mathsf{T}}\Sigma_{\text{vm}}{}^{-1}(\mathbf{x}-\mathbf{x}_{\text{vm}})\right)}. \tag{22}$$

By using these weights (to determine if an *individual* virtual guide is active in the first place), as well as the probability $p(n;\mathbf{x}_{\text{r}},s^n_{\text{vm}})$ (to determine the relative weighting *between all the guides*), the resultant force becomes

$$\mathbf{F}_{\text{res}} = \sum_{n=1}^{N} h(\mathbf{x}_{\text{r}};s^n_{\text{vm}})p(n;\mathbf{x}_{\text{r}},s^n_{\text{vm}})\mathbf{F}^n_{\text{vm}}. \tag{23}$$

The overall effect of this weighting scheme is the following: if the robot end-effector is not close to any guide, it is simply in zero-gravity mode, as none of the guides exerts a force to pull the end-effector towards the guide[7]. As soon as the robot end-effector approaches one of the guides, it starts exerting a force, and the end-effector is pulled toward the guide. The relative scaling between the guides is determined by the relative probability that the guide is responsible.

6 Refining Guides

To *modify* a guide we can exploit the incremental EM presented in (Calinon, 2007). After the user provides a new demonstration, the incremental clustering detects if the demonstration belongs to an existing guide, see Section 6.2. In this case, the demonstration is used to incrementally train the GMM which gets updated with the new data. Otherwise the demonstration is used to create a new guide, see Fig. 9.

6.1 Incremental GMM Estimation

The idea is to adapt the classic EM algorithm by splitting the part related to the old data from the part dedicated to the newly demonstrated data (Calinon, 2007). The update of the model is done under the assumption that the set of posterior probabilities $\{p(k|\zeta_m)\}_{j=1}^{M}$ remains the same when the new data $\{\tilde{\zeta}\}_{m=1}^{\tilde{M}}$ is used to update the model, this is called *data coherency constraint*. This assumption is true only if the new data is close to the trained model. This means that it is necessary to determine if the new data belongs or not to an already trained GMM (as anticipated, we will address this problem in the next section). Thus, the model is first created using the classic EM algorithm. Starting from an initial estimation[8] of the GMM with parameters $\{\pi^0_k,\mu^0_k,\Sigma^0_k\}_{k=1}^{K}$, at each step the following two steps are performed until a stop criterion is met:

E-step:

$$p^{t+1}_{k,m} = \pi^t_k \mathcal{N}(\zeta_m;\mu^t_k,\Sigma^t_k), \tag{24}$$

$$E^{t+1}_k = \sum_{m=1}^{M} p^{t+1}_{k,m}. \tag{25}$$

M-step:

$$\pi^{t+1}_k = \frac{E^{t+1}_k}{M}, \tag{26}$$

$$\mu^{t+1}_k = \frac{\sum_{m=1}^{M} p^{t+1}_{k,m}\zeta_m}{E^{t+1}_k}, \tag{27}$$

$$\Sigma^{t+1}_k = \frac{\sum_{m=1}^{M} p^{t+1}_{k,m}(\zeta_m-\mu^{t+1}_k)(\zeta_m-\mu^{t+1}_k)^{\mathsf{T}}}{E^{t+1}_k}. \tag{28}$$

The EM algorithm stops after a certain number of iterations T when

$$\frac{\mathscr{L}^{t+1}}{\mathscr{L}^t} - 1 < \mathscr{C}, \tag{29}$$

with the $\mathscr{L}$ defined in (11)[9]. The resulting GMM is completely defined by the set of parameters $\theta = \{\pi^T_k,\mu^T_k,\Sigma^T_k,M\}_{k=1}^{K}$.

When a new demonstration is provided by the user, $\tilde{T}$ steps are performed to update the model with the new data $\tilde{\zeta}$ with initial condition given by the previous model $\{\tilde{\pi}^0_k,\tilde{\mu}^0_k,\tilde{\Sigma}^0_k,\tilde{E}^0_k\}_{k=1}^{K} = \{\pi^T_k,\mu^T_k,\Sigma^T_k,E^T_k\}_{k=1}^{K}$ with $\tilde{E}^0_k = \tilde{\pi}^0_k M$.

The EM algorithm can be rewritten as:

E-step:

$$\tilde{p}^{t+1}_{k,m} = \tilde{\pi}^t_k \mathcal{N}(\tilde{\zeta}_m;\tilde{\mu}^t_k,\tilde{\Sigma}^t_k), \tag{30}$$

$$\tilde{E}^{t+1}_k = \sum_{m=1}^{\tilde{M}} \tilde{p}^{t+1}_{k,m}. \tag{31}$$

M-step:

$$\tilde{\pi}^{t+1}_k = \frac{\tilde{E}^0_k + \tilde{E}^{t+1}_k}{M+\tilde{M}}, \tag{32}$$

$$\tilde{\mu}^{t+1}_k = \frac{\tilde{E}^0_k\tilde{\mu}^0_k + \sum_{m=1}^{\tilde{M}} \tilde{p}^{t+1}_{k,m}\tilde{\zeta}_m}{\tilde{E}^0_k + \tilde{E}^{t+1}_k}, \tag{33}$$

$$\tilde{\Sigma}^{t+1}_k = \frac{\tilde{E}^0_k(\tilde{\Sigma}^0_k + (\tilde{\mu}^0_k-\tilde{\mu}^{t+1}_k)(\tilde{\mu}^0_k-\tilde{\mu}^{t+1}_k)^{\mathsf{T}})}{\tilde{E}^0_k + \tilde{E}^{t+1}_k}, \tag{34}$$

$$+ \frac{\sum_{m=1}^{\tilde{M}} \tilde{p}^{t+1}_{k,m}(\tilde{\zeta}_m-\tilde{\mu}^{t+1}_k)(\tilde{\zeta}_m-\tilde{\mu}^{t+1}_k)^{\mathsf{T}}}{\tilde{E}^0_k + \tilde{E}^{t+1}_k}. \tag{35}$$

Also in this case, the number of iterations $\tilde{T}$ is determined by the stop criterion defined in (29).

[7] We call the resulting guides "Soft Guides"

[8] In practice, the initial estimation is frequently performed using the K-means clustering algorithm which defines the initial values for the priors, means and covariance matrices.

[9] The threshold $\mathscr{C} = 0.01$ is used in our case.

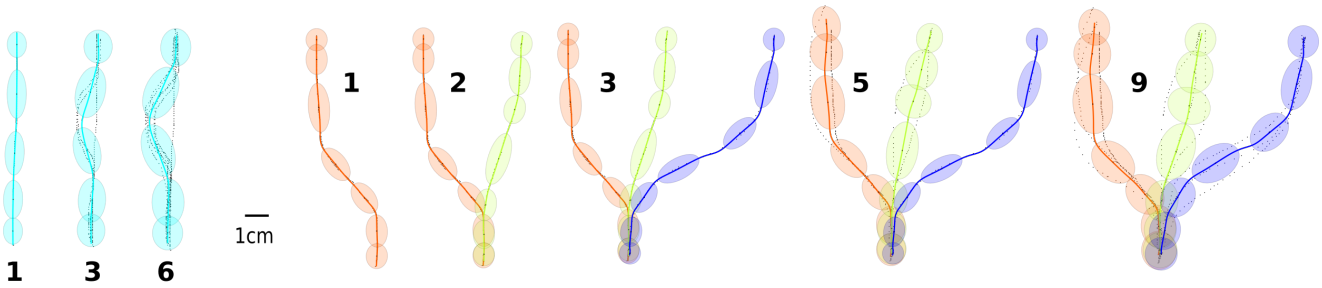

Fig. 9: Left: With the incremental training of GMM it is possible to iteratively modify an existing virtual guide. The number indicates the number of demonstrated trajectories used to incrementally train the guide. Right: Incremental training including incremental clustering, as explained in Section 6.2.

6.2 Incremental clustering

When the user demonstrates a new trajectory, we have to automatically detect if the new data belongs to one of the guide that has been previously created or if can be used to create a new one. This is necessary given the constraints on the data coherency of the proposed incremental EM. When the user demonstrates a new trajectory, the new data $\tilde{\zeta}$ is used to create a new GMM with the following parameters $\theta_{new} = \{\pi_k, \mu_k, \Sigma_k, M\}$. Since these parameters are the result of a set of EM steps, the associated likelihood represents the maximum likelihood, i.e. $L(\theta_{new}|\tilde{\zeta}) = L(\theta_{ML}|\tilde{\zeta})$. We can use the maximum likelihood $L(\theta_{ML}|\tilde{\zeta})$ as a baseline to select which GMM best fits the data $\tilde{\zeta}$. To perform the comparison we use the relative likelihood (Held and Bov, 2013) expressed as:

$$\hat{L}(\theta_n|\tilde{\zeta}) = \frac{L(\theta_n|\tilde{\zeta})}{L(\theta_{ML}|\tilde{\zeta})}, \forall\, n = 1..N, \tag{36}$$

where $L(\theta_n|\tilde{\zeta})$ represents the likelihood of an existing model n given the new demonstrated data $\tilde{\zeta}$. In particular, we have $1 \geq \hat{L}(\theta_n) \geq 0$ and $\hat{L}(\theta_{ML}) = 1$; because of this property, the relative likelihood is also called the normalized likelihood. The same expression can be computed using the log-likelihood, i.e. $\hat{\mathscr{L}}(\theta_n) = \log(\hat{L}(\theta_n)) = \mathscr{L}(\theta_n) - \mathscr{L}(\theta_{ML})$ where for the log-likelihood we have $0 \geq \hat{\mathscr{L}}(\theta_n) > -\inf$ with $\hat{\mathscr{L}}(\theta_{ML}) = 0$. For simplicity, we omitted the data set $\tilde{\zeta}$ since is the same in each comparison. We can compute the relative likelihood $\hat{L}(\theta_n)$ for each existing GMM. As proposed in (Held and Bov, 2013), we can select the model to update by using the following categorization based on the relative likelihood and a threshold c:

$$1 \geq \hat{L}(\theta_n) > c. \tag{37}$$

The threshold c can be selected arbitrarily. For example, we could categorize the likelihood as:

$$1 \geq \hat{L}(\theta_n) > \frac{1}{3} \quad \theta_n \text{ very plausible}, \tag{38}$$

$$\frac{1}{3} \geq \hat{L}(\theta_n) > \frac{1}{10} \quad \theta_n \text{ plausible}, \tag{39}$$

$$\frac{1}{10} \geq \hat{L}(\theta_n) \geq 0 \quad \theta_n \text{ not plausible}. \tag{40}$$

However, such a pure likelihood approach to inference has the disadvantage that the threshold c is somewhat arbitrarily chosen. The candidate model to be updated with the new incoming data is chosen in the interval given by (38). Between all the models that satisfy this inequality, we select the model with the maximum $\hat{L}(\theta_n)$. If none of the available models satisfy (38), the model θ_{ML} is used to create a new guide. This method requires the creation of a new GMM each time new data is provided. By creating a new model, we can keep track of the updates, meaning that the user can, at any time, revert a guide to its original shape. The advantages of this method are: it is easy to implement, fast and configurable due to the parameter c, does not require storing the previous data (only the GMM parameters are stored). The main drawbacks are: the selection and the significance of c and the necessity to create a new GMM that could not be used.

7 Experimental Evaluation

The following experiment was conducted on a 3-DOF ISY-BOT[10] comanipulation robot with a gripper, see Fig. 11. The general task was to use the robot and the library of virtual guides[11] to simulate pick and place operations. For the virtual guide assistance, the stiffness was set as $K = kI$ with $k = 10000\,N/m$ and the damping as $B = bI$ with $b = 400\,N/ms^{-1}$.

[10] http://www.isybot.com

[11] The code used to generate and interact with the library of virtual guides is available at https://github.com/graiola/virtual-fixtures

To create the virtual guides we used the incremental training presented in Section 6 with a fixed number of 10 Gaussians per model. Each demonstration was composed by $M = 2000$ samples $\{\zeta_m = [x(t_m), y(t_m), z(t_m)]\}_{m=1}^{M}$ recorded at 100 Hz. The phase s for each time step was computed with dynamic time warping and the mapping $s(t_m) = (t_m - t_1)/(t_M - t_1)$, as explained in Section 4.1.1.

7.1 User Study

We designed the study to observe: (1) how novice users perceived the virtual guide assistance with multiple guides, (2) to determine if creating new virtual guides with the library is intuitive and comfortable. We recruited 20 participants (with age between 22 and 33 years old, 7 females). Twelve participants stated they had prior experience with robots. We divided the user study in 4 sessions:

1 The user performs a pick and place task without the guides. (pp_1)
2 The user performs a pick and place task with multiple *default* guides active. (pp_2)
3 The user is *trained* on how to use the library of guides, afterwards the user is able to create his *personal* set of guides. (tr)
4 The user performs the pick and place task with the guides created in the previous session. (pp_3)

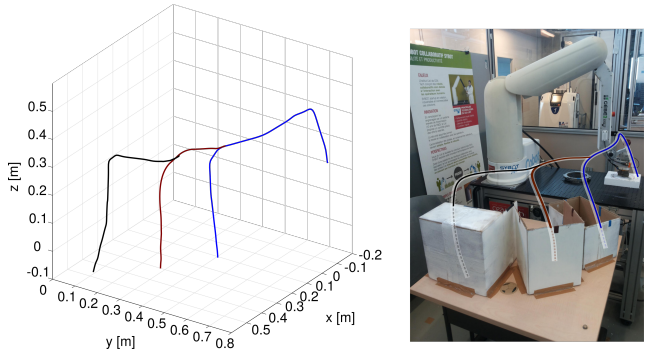

Fig. 10: Default virtual guides created by the expert user. Left: The colored lines represent the mean of the GMMs. Right: Virtual guides in the robot workspace.

The *default* guides for the session pp_2 were generated by an expert user with the library of virtual guides, see Fig. 10.

Four hypotheses were tested:

- H1: Virtual guides assistance improves task's performances, in terms of time and collisions occurrences.
- H2: Virtual guides assistance is more helpful when the task requires higher level of attention.
- H3: Virtual guides assistance is perceived as useful by the users.
- H4: It is intuitive and comfortable for novice users to create new virtual guides.

All participants were asked to perform the four sessions. The sessions $pp_{1,2}$ were presented in a randomized order to avoid training effects, while pp_3 was always presented after the session tr. At the beginning of $pp_{1,2}$, the participants were able to familiarize with the system.

7.2 Task explanation

The task in $pp_{1,2,3}$ consisted in taking 6 discs from the robot's workstation and insert them inside specific boxes identified with 3 different colors: blue, brown and black. For each box there were two discs with a piece of tape of the same color, see Fig. 11. The objective of the task was to place the discs in the associated box trying to minimize the time in respect of two constraints:

- The participant had to avoid collisions between the robot and the boxes.
- The discs had to be placed gently inside the boxes (it was not possible to drop the discs in the boxes to save time).

The boxes were disposed to obtain an increasing difficulty in terms of distance and accessibility ranging from the easiest (blue) to the hardest (black), see Fig. 11.

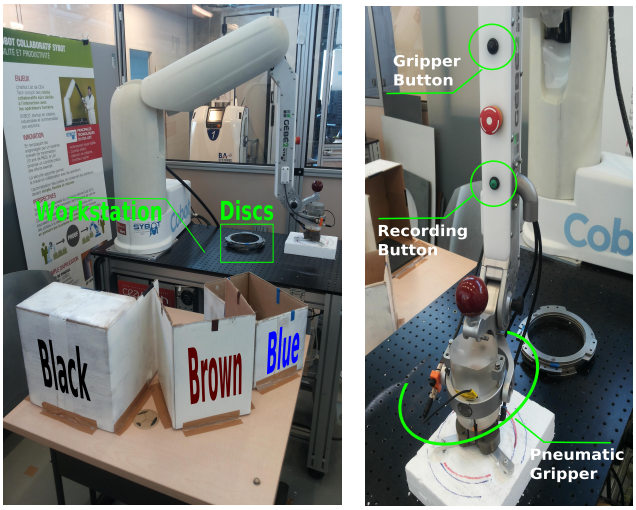

Fig. 11: Setup for the user study (left) and buttons used for the experiment (right). The upper button was used to hold and release the discs with the pneumatic gripper, while the lower button was used to start and stop the recording of the demonstrations.

We measured the total task time (T_j) necessary to complete the single session pp_j with $j = 1,2,3$ (the time T_j was

taken starting from the pick of the first disc and ending when the last disc was placed) and the pick and place time for each disc and session ($t_{i,j}$) with $i = 1,..,6$ (which leads to $18 = 6x3$ measures for each participant). The total time T_j differs from the sum over the single times $\sum_{i=1}^{6} t_{i,j}$ because it includes the time necessary for the user to pick the disc with the robot and to bring the robot back to the workstation after each placing.

In session tr the experimenter explained to the participants how to interact with the system, see Fig. 11. The participants were able to create their own guides in order to execute the task in session pp_3. During this session, the participants were allowed to ask for help from the experimenter. No time was recorded in tr.

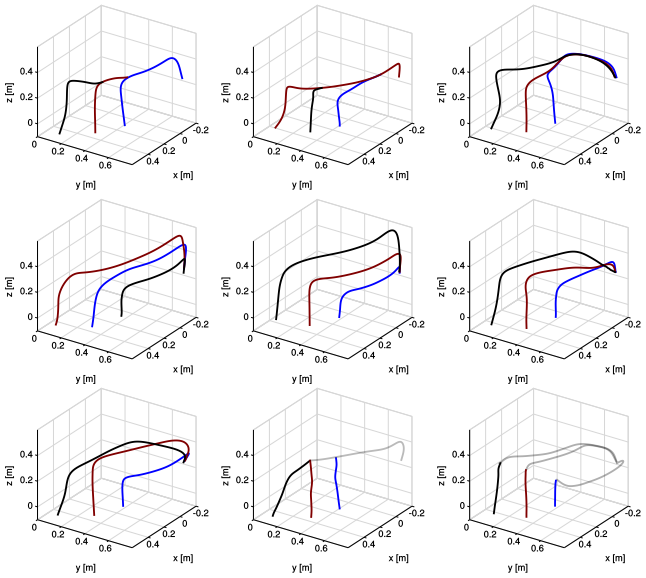

Fig. 12: Different approaches to the guides creation done by eight of our participants. Blue, brown and black curves are the guides created to place the disc inside the respective box. Gray curves are extra guides created to help connecting the guides. For comparison in the upper-left corner there are the guides created by the expert user.

Resulting guides from eight different users are shown in Fig. 12. At the end of $pp_{1,2,3}$ the participants answered a post-condition survey focusing on the usage experience with the virtual guides on the form of a Likert-scale survey with a rating from 1 to 7, with 1 as strong disagreement and 7 as strong agreement, see Table 1. For session tr the participants answered to a different survey focusing on the creation of the virtual guides, see Table 2.

7.3 Results

To validate our hypothesis we measured:

- Total task time T_j for each session $pp_{1,2,3}$ and pick and place time $t_{i,j}$ for each disc and session. Both are used to validate H1 and H2.
- Observed collisions, to validate H1 and H2.
- Survey results for sessions $pp_{1,2,3}$ to validate H3.
- Survey results for session tr to validate H4.

We performed a repeated-measure ANOVA (Girden, 1992) on T_j, $t_{i,j}$ and on the survey, on three factors: (1) the sessions pp_j with $j = 1,2,3$, (2) the difficulty, represented by the three boxes (blue; brown; black) and (3) the repetitions for each box $(r_1; r_2)$. Posthoc analyses were performed with Tuckey's HSD test (Abdi and Williams, 2010). For the collisions we observed that the participants collided with the boxes only during $pp_{1,3}$. For this reason, we performed Fisher-exact test between $pp_{1,3}$ on the number of participants that did at least one collision during the task and those that did not collide during the task. The significance threshold was set to $p < 0.05$.

7.3.1 Time Analysis

Effects of sessions on time:
We found a statistically significant difference $(p = .0023)$ between the sessions $pp_{1,2,3}$ on the total time (T) (Fig. 13, Left). Posthoc analysis shows that $pp_{1,2}$ and $pp_{1,3}$ are statistically different $(p = .005, p = .005)$. In pp_1 (No Guides) the participants were slower than in $pp_{2,3}$ (Default and Personal Guides). In addition, we found a statistically significant difference $(p = .00076)$ between the sessions on the pick and place time (t) (Fig. 13, Right). Also in this case, the posthoc analysis shows that $pp_{1,2}$ and $pp_{1,3}$ are statistically different, with $(p = .004, p = .001)$ respectively. We found again that in pp_1 the participants were slower than in $pp_{2,3}$. These two results enlighten that the virtual guides reduced the time to complete the task (both total time and pick and place time for each disc). This validates H1. Moreover we found that there is not statistical difference between the execution time with default and personal guides. This indicates us that the users were able to create guides that were as efficient as the default guides created by the expert user. This is an indication that H4 may be true, which we further discuss in Section 7.3.3.

Effects of the difficulty on time:
As pointed out in 7.2, the difficulty related to the disc insertion is different between the boxes $(p < .001)$. This can be seen in Fig. 14. Even if not statistically relevant, we reported also the t related to each box and each session. We can observe that the disc insertion for the black box requires more time without guides, this can be explained with the distance of the box from the workstation and with its disposition that does not facilitate the disc insertion. Instead, with the guides the time seems to increase linearly, meaning that

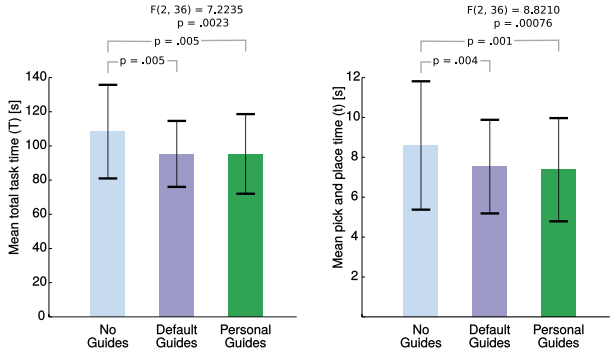

Fig. 13: Left: Mean of the total task time (T). Right: Mean of pick and place time (t).

the box disposition does not affect the insertion but only the distance does. These results support H2.

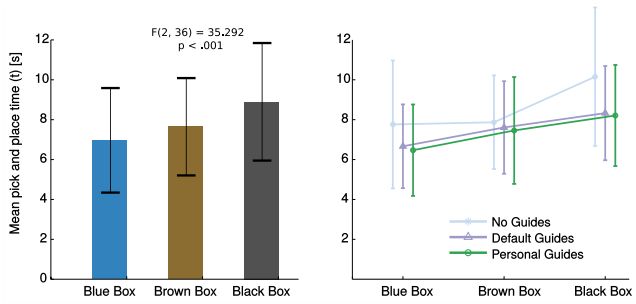

Fig. 14: Left: Mean of t for each box. Right: Mean of t for each box and each session.

Effects of repeated disc insertions on time:
We found a statistically significant effect $(p < .001)$ of the repeated insertions on t, see Fig. 15. The second repetition is shorter than the first. This represents a training effect on repeated disc insertions. However, we find a statistically significant interaction effect $(p = .047)$ between the sessions and the repetition. Posthoc analysis shows that in pp_1 there is no statistical difference between the two repetitions but in $pp_{2,3}$ the second repetition is shorter than the first one, with $(p = .047, p = .012)$ respectively. This informs us that, with guides, there is a training effect: repetitive use of virtual guides can improve the user performances. This is not necessary to prove H1 but it is a factor to take into account when using virtual guides.

7.3.2 Collisions Analysis

For the collisions, we measured that the participants collided with the boxes only during $pp_{1,3}$. In pp_2, the collisions were not possible due to the guides created by the expert user

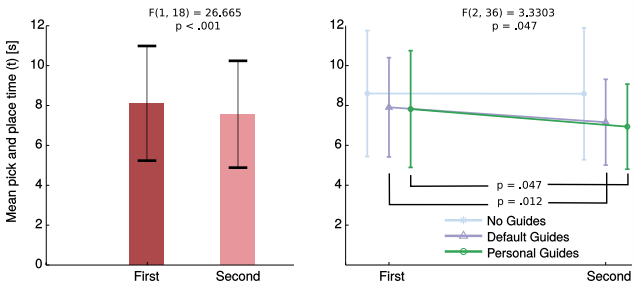

Fig. 15: Left: Mean of t for the two repetitions. Right: Mean of t for the two repetitions and each session.

(Fig. 10). In pp_3, collisions occurred because some participants did not proof-test their guides. For the collision, we found a statistical difference $(p = 0.0001)$ between pp_1 and pp_3: in pp_1, 18 of the 20 participants had at least one collision during the task, while in pp_3 only 6 of the 20 participants did. This indicates that using virtual guides leads to a safer task execution (Fig. 16), which goes in the direction of H1. Another observation could be done on the number of collisions for each box. As shown in Fig. 16, the majority of collisions occurred with the black box when the guides were not available. When the guides are used, the number of collisions with the black box reduces drastically (respectively 0 collisions with default guides and 1 collision with personal guides). This last result goes in direction of H2. The higher number of collisions with the blue box when the personal guides are used can be explained with the fact that the participants often started to create a guide for the blue box; this lead to an higher number of mistakes since it was their first training trial with the system.

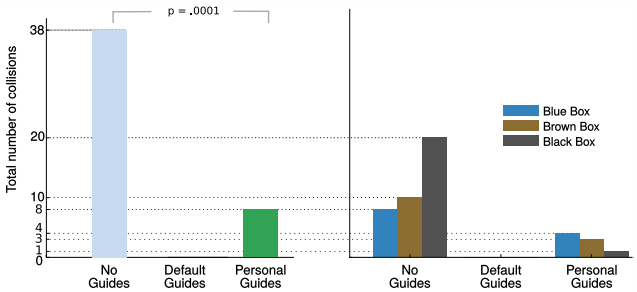

Fig. 16: Left: Total number of collisions by session. Right: Total number of collisions for each box and session.

7.3.3 Survey on Pick and Place

Table 1 shows the results of the survey. From it we can observe the following:

	$pp1$		$pp2$		$pp3$			
Question	**Mean**	**SD**	**Mean**	**SD**	**Mean**	**SD**	**F(2, 38)**	**P-value**
1) Do you think the task was easy to perform?	5.0	1.45	**5.75**	1.21	**5.85**	1.0	3.6157	**0.03652**
2) Do you think that you performed well the task?	4.6	1.57	**5.8**	0.83	**5.45**	1.1	7.4660	**0.00184**
3) Do you think the robot was helpful during the task execution?	4.3	1.52	**5.45**	1.54	**5.75**	1.02	10.298	**0.00027**
4) You felt comfortable with the robot while performing the task:	5.25	1.55	5.5	1.1	5.65	1.2	0.59262	0.55791
5) You felt stressed to use the robot while performing the task:	2.35	1.35	1.75	0.85	2.3	1.56	1.9334	0.15862
6) Do you think the robot is easy to work with:	4.85	1.46	**5.6**	1.35	**5.55**	1.0	2.6567	**0.08319**
7) Did you feel you had to put physical effort to perform the task:	2.75	1.65	3.25	1.8	2.8	1.36	1.4790	0.24068
8) Did you feel you performed the task precisely?	4.25	1.52	**5.8**	0.89	**5.25**	0.96	13.048	**0.00005**
9) Did you feel constrained by the robot during the experience?	2.95	1.64	4.25	1.77	3.3	1.69	4.4915	0.01774

Table 1: Survey results for the three pick and place sessions.

1 The task was perceived as easier to perform when using guides (both with default and personal guides).
2 Users thought that they performed better the task when using guides. Particularly better using the default guides. This can be verified with the collisions (no collisions using default guides, few collisions using personal guides). Moreover, the default guides were more precise since they were created by an expert user, see Fig. 10, while the personal guides were created in a little time by novice users.
3 Participants felt the robot was more helpful to perform the task when using guides. No relevant difference between default and personal guides.
4 Participants felt more comfortable with their own guides. In this case the result is not statistically relevant.
5 Participants felt less stressed when using the default guides, but more stressed when using their own guides. Again, this could be explained with the number collisions occurred during $pp3$. In this case we have a weak relevance.
6 Participants felt that was easier to work with the robot when the guides were active.
7 Participants perceived that they had to put more physical efforts to perform the task with the default guides. This could be explained by the fact that the controller generates a correction when the user tries to move away from the guide. This effect, reduces the naturalness of the switching when multiple guides are used. To solve this, it could be useful to measure or estimate user's external inputs, such as the force or the velocity applied on the end-effector, to facilitate the switch. Another possible reason for this result, is related to the fact that during the experiments some participants did not have a clear vision of where the guides were placed. During the task execution, some participants, instead of moving the robot along the guide, tried to move the robot where they wanted. Even if not statistically relevant, this could be interpreted as a clear evidence that some sort of visualization for the guides is needed.
8 Participants felt that they performed the task more precisely when using guides.

Question	**Mean**	**SD**
1) I believe that creating the new guides was:		
- Intuitive	5.35	1.31
- Comfortable	4.85	1.04
- Physically demanding	3.1	2.02
- Cognitively demanding	3.9	1.71
2) I believe that to perform the task I should use:		
- NO Guides at all	3.15	1.56
- ONE Guide	3.1	1.74
- MULTIPLE Guides	5.75	1.5
3) I believe that the guide(s) I created reflected what I demonstrated	5.7	1.17
4) I believe that the guide(s) I created was(were) precise	5.0	1.25

Table 2: Survey results for the training session.

9 The participants felt more constrained when using the default guides. This can be explained by the fact that is easier to feel one's own guides than guides created by another person.

By looking at the results highlighted in Table 1 for the questions $1,2,3,6,8$ we can confirm that virtual guide assistance is perceived as useful by the users, which validates H3.

7.3.4 Survey Training

From (Table 2) we can see that participants felt that creating the virtual guides was quite intuitive and comfortable (question 1). For the selected task multiple guides were felt as necessary (question 2). Moreover, participants felt that the guides they created effectively reflected what they demonstrated (question 3) and were enough precise (question 4). With these results we can validate H4.

8 Conclusions

The development of robotics tools such as virtual guides can be very useful to improve human performances in industrial tasks that can not be completely automatized. Robots

possess characteristics such as precision, strength and accuracy that can be exploited in co-manipulation tasks by using the virtual guiding assistance. In this paper, we presented a novel way to create virtual guides; we developed an intuitive and easy way to program them through kinesthetic teaching by using Gaussian Mixture Models. Furthermore, the incremental training of GMM enables the user to refine the guides iteratively with the possibility to be assisted by the virtual guide during the refining process. We also defined a controller that allows the user to use multiple virtual guides in parallel, and selects which guide is responsible for the task execution, based on the variance estimated by the GMM. Together, this constitutes a *library* of virtual guides that enables the user to create, modify and use multiple guides. Finally, we studied the utility of virtual guides with an industrial task and concluded that virtual guides improve the human performances in terms of time and collisions, and they can relieve the workload from the user. In future work we will explore the possibility to exploit the uncertainty given by the probabilistic model to adapt the stiffness of the guide (Medina et al, 2012; Calinon et al, 2014) and add a way to visualize the virtual guides in order to increase the user's immersion in the robot's workspace (Rosenberg, 1993).

Acknowledgements This project has received funding from DIGITEO digiteo (www.digiteo.fr)

FS received funding from the European Commission through the project H2020 AnDy (GA no. 731540)

References

Aarno D, Ekvall S, Kragic D (2005) Adaptive virtual fixtures for machine-assisted teleoperation tasks. In: ICRA, pp 897–903

Abbott JJ (2005) Virtual fixtures for bilateral telemanipulation. PhD thesis, Johns Hopkins University

Abdi H, Williams LJ (2010) Tukeys honestly significant difference (hsd) test. Encyclopedia of Research Design Thousand Oaks, CA: Sage pp 1–5

Amor HB, Neumann G, Kamthe S, Kroemer O, Peters J (2014) Interaction primitives for human-robot cooperation tasks. In: Robotics and Automation (ICRA), 2014 IEEE International Conference on, IEEE, pp 2831–2837

Becker BC, Maclachlan RA, Lobes LA, Hager GD, Riviere CN (2013) Vision-based control of a handheld surgical micromanipulator with virtual fixtures. IEEE Trans Robot 29(3):674–683

Bettini A, Marayong P, Member S, Lang S, Okamura AM, Hager GD (2004) Vision assisted control for manipulation using virtual fixtures. In: International Conference on Intelligent Robots and Systems (IROS), pp 1171–1176

Bowyer SA, y Baena FR (2013) Dynamic frictional constraints for robot assisted surgery. In: World Haptics Conference (WHC), 2013, pp 319–324, DOI 10.1109/WHC.2013.6548428

Bowyer SA, Davies BL, y Baena FR (2014) Active constraints/virtual fixtures: A survey. IEEE Transactions on Robotics 30(1):138–157, DOI 10.1109/TRO.2013.2283410

Boy ES, Burdet E, Teo CL, Colgate JE (2007) Investigation of motion guidance with scooter cobot and collaborative learning. IEEE transactions on robotics 23(2):245–255

Calinon S (2007) Incremental learning of gestures by imitation in a humanoid robot. In: In Proceedings of the 2007 ACM/IEEE International Conference on Human-Robot Interaction, pp 255–262

Calinon S, Guenter F, Billard A (2007) On learning, representing and generalizing a task in a humanoid robot. IEEE Transactions on Systems, Man and Cybernetics, Special issue on robot learning by observation, demonstration and imitation 37(2):286–298

Calinon S, Bruno D, Caldwell DG (2014) A task-parameterized probabilistic model with minimal intervention control. In: Proc. IEEE Intl Conf. on Robotics and Automation (ICRA), Hong Kong, China, pp 3339–3344

Colgate JE, Peshkin MA, Klostermeyer SH (2003) Intelligent assist devices in industrial applications: a review. In: IROS, pp 2516–2521

David O, Russotto FX, Simoes MDS, Measson Y (2014) Collision avoidance, virtual guides and advanced supervisory control teleoperation techniques for high-tech construction: Framework design. Automation in Construction 44:63–72

Davies B, Jakopec M, Harris SJ, Baena FRY, Barrett A, Evangelidis A, Gomes P, Henckel J, Cobb J (2006) Active-constraint robotics for surgery. Proceedings of the IEEE 94(9):1696–1704, DOI 10.1109/JPROC.2006.880680

Dumora J (2014) Contribution à linteraction physique homme-robot: application à la comanipulation dobjets de grandes dimensions. PhD thesis, Montpellier 2

Ewerton M, Maeda G, Kollegger G, Wiemeyer J, Peters J (2016) Incremental imitation learning of context-dependent motor skills. In: Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on, IEEE, pp 351–358

Girden ER (1992) ANOVA: Repeated measures. 84, Sage

Held L, Bov DS (2013) Applied Statistical Inference: Likelihood and Bayes. Springer Publishing Company, Incorporated

Hermann M, Pentek T, Otto B (2016) Design principles for industrie 4.0 scenarios. In: 2016 49th Hawaii International Conference on System Sciences (HICSS), IEEE, pp 3928–3937

Ho SC, Hibberd RD, Davies BL (1995) Robot assisted knee surgery. IEEE Engineering in Medicine and Biology Magazine 14(3):292–300, DOI 10.1109/51.391774

Joly L, Andriot C (1995) Imposing motion constraints to a force reflecting telerobot through real-time simulation of a virtual mechanism. In: Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on, vol 1, pp 357–362 vol.1, DOI 10.1109/ROBOT.1995.525310

Kuang A, Payandeh S, Zheng B, Henigman F, MacKenzie C (2004) Assembling virtual fixtures for guidance in training environments. In: Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2004. HAPTICS '04. Proceedings. 12th International Symposium on, pp 367–374, DOI 10.1109/HAPTIC.2004.1287223

Lee D, Ott C (2011) Incremental kinesthetic teaching of motion primitives using the motion refinement tube. Autonomous Robots 31(2-3):115–131

Li M, Okamura AM (2003) Recognition of operator motions for real-time assistance using virtual fixtures. In: In Proc. 11th Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems, pp 125–131

Lin HC, Marayong P, Mills K, Karam R, Kazanzides P, Okamura AM, Hager GD (2006) Portability and applicability of virtual fixtures across medical and manufacturing tasks. In: IEEE International Conference on Robotics and Automation, pp 225–340

Marayong P, Li M, Okamura AM, Hager GD (2003) Spatial motion constraints: theory and demonstrations for robot guidance using virtual fixtures. In: ICRA, IEEE, pp 1954–1959

Medina JR, Lee D, Hirche S (2012) Risk-sensitive optimal feedback control for haptic assistance. In: Robotics and Automation (ICRA), 2012 IEEE International Conference on, IEEE, pp 1025–1031

Mollard Y, Munzer T, Baisero A, Toussaint M, Lopes M (2015) Robot programming from demonstration, feedback and transfer. In: Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, pp 1825–1831, DOI 10.1109/IROS.2015.7353615

Nolin JT, Stemniski PM, Okamura AM (2003) Activation cues and force scaling methods for virtual fixtures. In: in Proc. 11th Int. Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems, pp 404–409

Pezzementi Z, Hager GD, Okamura AM (2007) Dynamic guidance with pseudoadmittance virtual fixtures. In: IEEE International Conference on Robotics and Automation, pp 1761–1767

Raiola G (2017) Co-manipulation with a library of virtual guides. PhD thesis, Université Paris-Saclay

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

Raiola G, Rodriguez-Ayerbe P, Lamy X, Tliba S, Stulp F (2015b) Parallel guiding virtual fixtures: Control and stability. In: Intelligent Control (ISIC), 2015 IEEE International Symposium on, IEEE, pp 53–58

Rosenberg L (1993) Virtual fixtures: perceptual tools for telerobotic manipulation. In: Proc. IEEE Virtual Reality International Sympsoium

Rozo L, Calinon S, Caldwell DG, Jimnez P, Torras C (2016) Learning physical collaborative robot behaviors from human demonstrations. IEEE Transactions on Robotics PP(99):1–15, DOI 10.1109/TRO.2016.2540623

Ryden F, Stewart A, Chizeck H (2013) Advanced telerobotic underwater manipulation using virtual fixtures and haptic rendering. In: Oceans - San Diego, 2013, pp 1–8

Sanchez Restrepo S, Raiola G, Chevalier P, Lamy X, Sidobre D (2017) Iterative virtual guides programming for human-robot comanipulation. In: IEEE International Conference on Advanced Intelligent Mechatronics (AIM)

Vakanski A, Mantegh I, Irish A, Janabi-Sharifi F (2012) Trajectory learning for robot programming by demonstration using hidden markov model and dynamic time warping. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 42(4):1039–1052, DOI 10.1109/TSMCB.2012.2185694

Wrede S, Emmerich C, Grünberg R, Nordmann A, Swadzba A, Steil J (2013) A user study on kinesthetic teaching of redundant robots in task and configuration space. Journal of Human-Robot Interaction 2(1):56–81

Yoon H, Wang R, Hutchinson S (2014) Modeling user's driving-characteristics in a steering task to customize a virtual fixture based on task-performance. In: Robotics and Automation (ICRA), 2014 IEEE International Conference on, pp 625–630, DOI 10.1109/ICRA.2014.6906920

Yu W, Alqasemi R, Dubey R, Pernalete N (2005) Telemanipulation assistance based on motion intention recognition. In: Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on, pp 1121–1126, DOI 10.1109/ROBOT.2005.1570266