A probabilistic approach for learning and adapting shared control skills with the human in the loop

Gabriel Quere, Freek Stulp, David Filliat, Joao Silverio

A probabilistic approach for learning and adapting shared control skills with the human in the loop

Gabriel Quere[1,2], Freek Stulp[1], David Filliat[2], and João Silvério[1]

[1]Institute of Robotics and Mechatronics, German Aerospace Center (DLR)
[2]U2IS, ENSTA Paris, IP Paris

***Abstract*— Assistive robots promise to be of great help to wheelchair users with motor impairments, for example for activities of daily living. Using shared control to provide task-specific assistance – for instance with the Shared Control Templates (SCT) framework – facilitates user control, even with low-dimensional input signals. However, designing SCTs is a laborious task requiring robotic expertise. To facilitate their design, we propose a method to learn one of their core components – active constraints – from demonstrated end-effector trajectories. We use a probabilistic model, Kernelized Movement Primitives, which additionally allows adaptation from user commands to improve the shared control skills, during both design and execution. We demonstrate that the SCTs so acquired can be successfully used to pick up an object, as well as adjusted for new environmental constraints, with our assistive robot EDAN.**

I. Introduction

Assistive robots promise to re-enable users with motor impairments to perform activities of daily living. Shared control, combining human input commands with an autonomous control system to achieve a common goal, empowers users with the ability to interact with their environment in a convenient manner, see [1], Ch. 43, for an overview. Shared control alleviates the many difficulties that arise from the direct control of a high degree-of-freedom (DoF) system – like our assistive robot EDAN [2], [3], see Fig. 1, *left* – such as cumbersome mode switches between position, orientation, gripper and wheelchair control [2].

In previous work we introduced Shared Control Templates (SCT) [4] with the aim to provide robust and legible assistance to users for successful task execution. Notably, SCTs were recently used at the Cybathlon Challenge[1] to successfully assist a motor-impaired pilot on EDAN to pick and place objects and pick and bite into an apple within 5 minutes. SCTs are represented as finite state machines, with each state having a number of components (see Section III-A), including *active constraints* [5], which restrict the robot task space to assist the user in completing a task.

Currently, designing SCTs requires robotics expertise. However, to fully exploit their potential, SCTs should be easy to both design and modify. In this work, we turn to probabilistic learning from demonstration to facilitate the design and adaptation of SCTs. Namely, we propose that the required constraints to manipulate objects are identified from demonstrated robot trajectories and modified via user commands when new task conditions arise. Leveraging user commands from external devices to correct assistive models locally is an interesting alternative to providing new kinesthetic demonstrations or haptic feedback, particularly in settings where motor-impaired users are involved. We thus follow a non-parametric approach using Kernelized Movement Primitives (KMP) [6] (Section III-B) to realize the learning and adaptation of SCT constraints with the human in the loop, aiming to make it intuitive and easy to use. We leverage the fact that KMPs encode the variance and correlations in the data, which we exploit to define active constraints in the form of generalized cylinders [7]. In addition, we build on recent results in uncertainty-aware, computationally-efficient motion primitive modulation [8] to adapt the learned constraints smoothly and locally, directly from user commands. The result is an interactive imitation learning approach [9] that does not require physical interactions with the robot for adaptation. In summary, the contribution of this paper is an approach to:

1) derive active constraints from a KMP model fitted on demonstrated end-effector trajectories (IV-A),
2) allow users to modify those learned constraints directly from user commands, adapting the assistance provided by the framework to new environmental constraints and requirements at runtime (IV-B).

Our approach is experimentally validated on EDAN by performing a picking task (Section V). We show that, using our approach, multiple able-bodied users are able not only to complete the task but to adapt the assistance provided by the robot to new situations. We further show that the required modulations are done in sensible time frames and with deformations of the original model to an expert user.

II. Related work

The topic of designing adaptive assistive behaviors has received attention in recent years. Mehr *et al.* [10] inferred end-effector constraints online while the user was performing a task. Broad *et al.* [11] propose corrections via natural language for planned motions. In Selvaggio *et al.* [12], users in a surgery setting can online generate active constraints and switch between them, with stiffness adaptation. While those works can be intuitive or modular, they lack the expressiveness and ease of design that learning approaches can provide.

[1]https://www.youtube.com/watch?v=EoER_5vYZsU

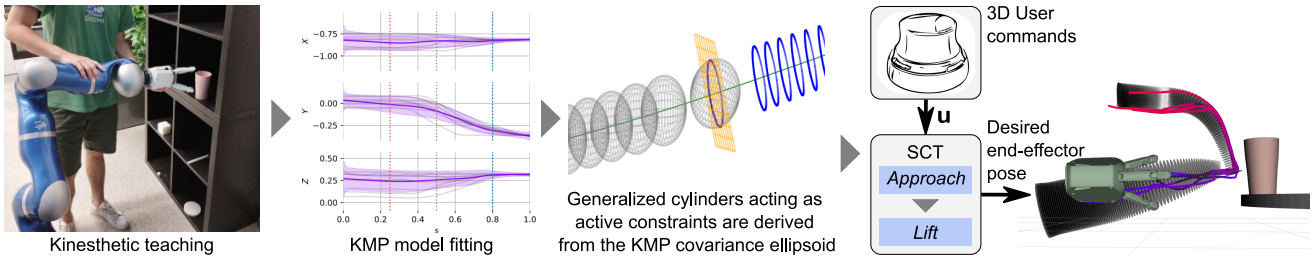

Fig. 1. Schema of the proposed approach to design Shared Control Templates (SCTs) with a probabilistic model: Kernelized Motion Primitives (KMP). A KMP is fitted to a set of demonstrated trajectories for the desired task. Active constraints are then derived from this model, and an SCT is built. Finally, the robot is constrained to successfully complete the task from user commands.

Learning from demonstration has been investigated as a promising way to intuitively encode assistance. SCT constraints were designed from a pre-defined set of geometric constraints extracted from demonstrations in [13]. Learning position-dependent input mappings with a conditional auto-encoder was proposed in [14]. Perez *et al.* [15] learned geometric constraints for a state machine but target supervised autonomy. Havoutis *et al.* [16] proposed a task-parameterized formulation taking an object-centered approach to assist teleoperators in handling ambiguities between local and remote workspaces. In Iregui *et al.* [17], a reconfigurable and adaptable approach is learned from demonstrations using Probabilistic Principal Component Analysis [18]. Unlike our approach, those works do not propose external modulation from human inputs.

In [19], Gaussian mixture models (GMMs) are used to encode constraints from demonstrations by using precision matrices as a proxy for control stiffness. Being based on GMMs, [16], [19] are not easily adaptable to new task conditions from human inputs at run-time, requiring new data when adaptation is needed. The non-parametric nature of KMPs facilitates the modulation of learned trajectories by external signals, particularly via approaches that take into account the uncertainty in the data when applying modulations [8]. Similar strategies can, in principle, be implemented using Gaussian processes (GPs) [20]. However, due to not modeling aleatoric uncertainties, GPs are less attractive to model constraints.

Behavior adaptation has also been studied without considering assistance: Losey *et al.* proposed in [21] to deform a trajectory during execution with haptic feedback and constrained optimization. Ahmadzadeh and Chernova [7] proposed to use generalized cylinders – surfaces with smoothly-varying cross-sections – to generate autonomous behavior from trajectories. Although they address adaptation, it is done in a similar fashion to [19], requiring new kinesthetic demonstrations or physical interactions with the robot. Instead of haptic feedback, due to our target users, we propose adaptation from user commands.

III. Background

A. Shared Control Templates (SCTs)

An SCT [4] is a finite-state machine tailored to shared control, where each state defines *input mappings*, *transition conditions* and *active constraints*.

Input mappings map user commands, typically low-dimensional, to manipulator end-effector velocities, aiming to let the user be in control of relevant task DoFs. They range from standard translational or rotational control, to, e.g., more complex rotation control around the tip of a grasped bottle when pouring. At present, EDAN mainly expects 3D commands $\boldsymbol{u} \in [-1, 1]^3$, generated by joystick or surface electromyography [2].

Transition conditions specify conditions that need to be fulfilled in each state of an SCT (e.g. end-effector poses with respect to objects, contact forces, user trigger) such that its states are executed sequentially.

Finally, SCTs use *active constraints* as regional constraints as defined by Bowyer et al. [5], which keep a desired end-effector pose within a restricted region of the task space. With SCTs, this is achieved by projecting at every time-step the target end-effector pose, computed from input commands, back into the allowed region, if constraints are violated. Examples of active constraints include limit tilting angles in a *pour* task, planar or volumetric primitive constraints (e.g. cones or funnel [13]) that restrict the end-effector position, helping the user control the robot to approach an object.

Through the definition of states and their components, SCTs define *shared control skills* that permit robots to complete tasks by leveraging user commands in principled ways. As an example, a *pick* skill can be defined by an SCT through states for approaching an object, and lifting this object, each one constrained with a generalized cylinder, as seen in Fig. 1, *right*. A *pour* skill would have states for approaching a cup with a grasped bottle, then tilting the bottle, thus pouring its contents into the cup. SCTs are time-independent, object-centric and assume a static world model during task execution.

B. Kernelized Movement Primitives (KMPs)

KMPs [6] are function approximators used in imitation learning to predict the value of an output variable $\boldsymbol{\xi} \in \mathbb{R}^D$ given observations of an input $\boldsymbol{s} \in \mathbb{R}^I$ from a set of end-effector trajectories. A KMP assumes that a *reference trajectory distribution* $\{\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n\}_{n=1}^N$, encoding the means, variations and correlations of $\boldsymbol{\xi}$, is available to model $\mathcal{P}(\boldsymbol{\xi}|\boldsymbol{s}_n)$, where $\boldsymbol{s}_{n=1,\dots,N}$ are N given inputs. In [6], $\boldsymbol{\mu}_n$, $\boldsymbol{\Sigma}_n$ are computed from a GMM. The expectation of the output variable is computed, for a test input $\boldsymbol{s}^*$, using:

$$\mathbb{E}[\boldsymbol{\xi}(\boldsymbol{s}^*)] = \boldsymbol{k}^* \left(\boldsymbol{K} + \lambda_1 \boldsymbol{\Sigma}\right)^{-1} \boldsymbol{\mu}, \tag{1}$$

where $\boldsymbol{k}^*$, $\boldsymbol{K}$ are evaluations of $\boldsymbol{s}^*$ and $\boldsymbol{s}_{n=1,\dots,N}$ using a kernel function $k(\boldsymbol{s}_i, \boldsymbol{s}_j)$ (see [6] for details), $\lambda_1 > 0$ is a regularization factor and $\boldsymbol{\mu} = \left[\boldsymbol{\mu}_1^\top, \dots, \boldsymbol{\mu}_N^\top\right]^\top$, $\boldsymbol{\Sigma} = \text{blockdiag}\left(\boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_N\right)$, a block-diagonal matrix. Moreover, the covariance of the output is given by:

$$\mathbb{D}\left[\boldsymbol{\xi}(\boldsymbol{s}^*)\right] = \alpha\left(\boldsymbol{k}^{**} - \boldsymbol{k}^*\left(\boldsymbol{K} + \lambda_2\boldsymbol{\Sigma}\right)^{-1}\boldsymbol{k}^{*\top}\right), \quad (2)$$

where α is a scaling factor, λ_2 is a regularization factor and $\boldsymbol{k}^{**}$ denotes the evaluation of the kernel function at $\boldsymbol{s}^*$, see [6] for details.

1) Model adaptation using via-points: As shown in [6], KMPs provide a principled way for trajectory modulation when new task conditions arise. Particularly, for a new input $\bar{\boldsymbol{s}}$, adding the pair $\{\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\Sigma}}\}$ to the reference trajectory distribution will ensure that the expected trajectory passes through a desired via-point $\bar{\boldsymbol{\mu}}$, provided that $\bar{\boldsymbol{\Sigma}}$ is small enough. While this ensures adaptation, it modifies the covariance profile through $\bar{\boldsymbol{\Sigma}}$ which can have undesirable effects if the variance (2) is used to represent task space constraints, leading to excessively constrained motions. Additionally it requires to invert the term $\boldsymbol{K} + \lambda\boldsymbol{\Sigma}$ every time a via-point is added, which has $O(n^3)$ complexity.

2) Model adaptation using null space actions: In recent work [8], the original KMP formulation was extended with a term that locally modulates the trajectory distribution, enabling a more efficient adaptation with $O(n^2)$ complexity, with no impact on the covariance profile. By defining a desired modulation $\hat{\boldsymbol{\xi}}$ at an input $\hat{\boldsymbol{s}}$ the resulting expectation is computed as:

$$\mathbb{E}[\boldsymbol{\xi}(\boldsymbol{s}^*)] = \boldsymbol{k}^*\boldsymbol{\Psi}\boldsymbol{\mu} + \left[\hat{\boldsymbol{k}}^* - \boldsymbol{k}^*\boldsymbol{\Psi}\hat{\boldsymbol{K}}\right]\hat{\boldsymbol{\xi}}, \quad (3)$$

where $\boldsymbol{\Psi} = \left(\boldsymbol{K} + \lambda_1\boldsymbol{\Sigma}\right)^{-1}$ and $\hat{\boldsymbol{k}}^*$, $\hat{\boldsymbol{K}}$ are evaluations of the kernel function at $\hat{\boldsymbol{s}}$, see [8] for details. Note that the first term in (3) corresponds to (1) and $\boldsymbol{\Psi}$ only needs to be computed once. The second term includes a projector $\left[\hat{\boldsymbol{k}}^* - \boldsymbol{k}^*\boldsymbol{\Psi}\hat{\boldsymbol{K}}\right]$ that modulates the original expectation only locally. This projector is referred in [8] as a *soft null space projector*, since, on the one hand, it is the solution to a least squares problem with null space and, on the other hand, it is a 'soft' projector, allowing $\hat{\boldsymbol{\xi}}$ to modify the original expectation in proportion to the data variance in the neighborhood of $\hat{\boldsymbol{s}}$.

IV. Proposed Approach

In this section we introduce an approach to extract active constraints from a KMP and adapt them with external user commands when new task conditions arise. The KMP expectation (1) is used to define a reference path, which, in combination with the covariance (2) at each point, allows the derivation of a generalized cylinder [7] that constrains the end-effector only along the directions orthogonal to the reference path (Section IV-A), see Fig. 1 for an overview. Subsequently, we use the approach from [8] to modulate the learned active constraints from user commands $\boldsymbol{u} \in \mathbb{R}^D$ by computing the term $\hat{\boldsymbol{\xi}}$ directly from $\boldsymbol{u}$. Thus, we act on the KMP expectation through the projector (3) and consequently on the reference path (Section IV-B). Figure 2 illustrates this concept. In order to mitigate undesired correlations between actions and modulate multiple points simultaneously we introduce extensions of [8] (Sections IV-C and IV-D).

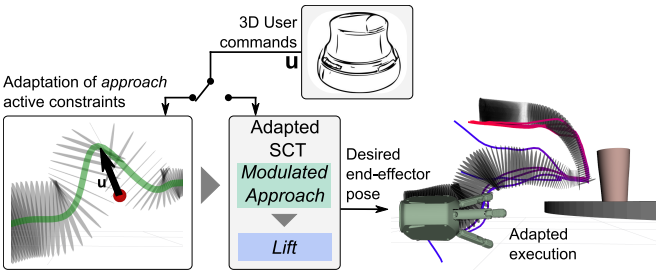

Fig. 2. If an SCT requires adaptation due to environmental changes or user preferences, a *correction mode* is entered, where the user can directly adapt the SCT with their commands. The adapted SCT is then executed.

A. Deriving active constraints from a KMP

Let us assume a set of H demonstrations with M datapoints each $\{\{\boldsymbol{s}_{m,h}, \boldsymbol{\xi}_{m,h}\}_{m=1}^M\}_{h=1}^H$ where $\boldsymbol{\xi}$ is the end-effector position and $\boldsymbol{s}$ is a normalized time variable aligned across all demonstrations using Dynamical Time Warping [22]. Querying a KMP model fit on such dataset with equally spaced inputs $\boldsymbol{s}^*_{i=1,\dots,N} \in [0, 1]$ provides a trajectory distribution over end-effector positions with associated means and covariance matrices $\{\boldsymbol{\mu}_i^*, \boldsymbol{\Sigma}_i^*\}_{i=1}^N$. We propose to use the KMP expectation (1), given by the means $\{\boldsymbol{\mu}_i^*\}_{i=1}^N$, as the directrix Γ of a generalized cylinder [7]. Additionally, for every point i along Γ, the covariance matrix $\boldsymbol{\Sigma}_i^*$ at a pre-defined variance threshold is treated as an ellipsoid. Its intersection with a plane passing through its center and perpendicular to the directrix – following the approach described in [23] – creates an ellipse ρ, see also Fig. 1, *center*. The set of ellipses computed at every point of Γ, $\{\rho_i\}_{i=1}^N$, forms a surface with smoothly-varying cross-sections – a generalized cylinder [7].

The generalized cylinder computed from the KMP is used as an active constraint acting in Cartesian space, by constraining the desired end-effector position within its volume. In practice, the closest $\boldsymbol{\mu}_i^*$ is identified, and the end-effector is projected into the cylinder spanned by ρ_i. A high variance at a specific section of the demonstrated trajectories provides motion freedom to the user, while a low variance constrains the user, e. g. at a specific grasp pose. Note that using a generalized cylinder representation guarantees unconstrained motion along the direction of Γ, unlike projecting on a full covariance ellipsoid which may have low variance along the direction of motion.

Given generalized cylinders extracted from demonstrations, we propose to adapt them with the user in the loop by acting directly on the underlying KMP representation.

B. KMP adaptation with user in the loop

Given the limitations of via-point-based KMP adaptation discussed in III-B (which are common to other kernel-based methods, e.g. GPs), we propose to act on the null space of the KMP (3) and adapt generalized cylinders directly from user commands $\boldsymbol{u}$. Since $\boldsymbol{u}$ is generally used to drive the robot end-effector we propose that, when wanting to modulate

the active constraints, the user triggers, at any instant t_0, a *correction mode*, where the shared control deactivates and commands are interpreted as desired modulations of the underlying KMP expectation (1) instead of control actions to the robot. In correction mode, the modulation term $\hat{\boldsymbol{\xi}}$ in (3) is computed directly from $\boldsymbol{u}$. For a given location $\hat{\boldsymbol{s}}$ in the input space, where the modulation is to be applied, we thus define a desired correction starting at t_0 and lasting until t_1 as an accumulation of user commands:

$$\hat{\boldsymbol{\xi}}(\hat{\boldsymbol{s}}) = \int_{t_0}^{t_1} \boldsymbol{u}(t)\, dt. \tag{4}$$

Equation (4) can be used generically to define modulations to be applied on a KMP through (3), from external user commands. This effect is shown in Fig. 2.

In our experiments (Section V) a 3D user command $\boldsymbol{u}$ is used, and is decoupled into two orthogonal components: the component tangent to the directrix, $\boldsymbol{u}_{\parallel}$, and the component perpendicular to it, $\boldsymbol{u}_{\perp}$. We propose to use $\boldsymbol{u}_{||}$ to select where the modulation is taking place, $\hat{\boldsymbol{s}}$, by moving along the directrix. At the same time, $\boldsymbol{u}_{\perp}$ is used to modulate the underlying KMP by acting on the directions orthogonal to the directrix. Note that, despite the decoupling, we have $\boldsymbol{u}_{\perp}, \boldsymbol{u}_{\parallel} \in \mathbb{R}^3$ as they are represented in the robot base. For simplicity, we keep the remaining of the discussion generic by using $\boldsymbol{u}$.

C. Decorrelating adaptations

Being constructed from full covariance matrices, the projector derived in (3) behaves in a way that takes into account the correlations in the training data. In such a case, a desired modulation on a subset of the task space DoFs, e.g. $\hat{\boldsymbol{\xi}} = [\hat{\xi}_1\ 0\ 0]^\top$ in our 3D case, may affect other DoFs. While this may be a desirable feature in exploration [8], it may lead to unintended effects in shared control scenarios. We therefore propose to decorrelate the result of applying (4). To this end, $\boldsymbol{u}$ is decomposed along its D components $\{u_j\}_{j=1}^{D}$ and (4) is computed for each component as $\hat{\boldsymbol{\xi}}_j$, with zero entries except at index j. The removal of correlations into the final modulation is then performed by computing

$$\mathbb{E}[\boldsymbol{\xi}(\boldsymbol{s}^*)] = \boldsymbol{k}^*\boldsymbol{\Psi}\boldsymbol{\mu} + \sum_{j=1}^{D} \bar{\boldsymbol{S}}_j \left[\hat{\boldsymbol{k}}^* - \boldsymbol{k}^*\boldsymbol{\Psi}\hat{\boldsymbol{K}}\right] \hat{\boldsymbol{\xi}}_j, \tag{5}$$

where $\bar{\boldsymbol{S}}_j = \text{blockdiag}(\boldsymbol{S}_{j,1}, \ldots, \boldsymbol{S}_{j,N})$ is a block-diagonal selection matrix formed of $D-$dimensional matrices $\boldsymbol{S}_{j,i}$, with zero entries except at (j,j) which is 1. In this manner, the uncertainty-aware modulation properties of the original projector (3) are kept (regions of low/high variance require more/less modulation commands to be modified) while ensuring that actions along one DoF do not affect other DoFs. This effect is shown in Fig 5. Alternatively, correlations could be removed directly from $\{\boldsymbol{\Sigma}_n\}_{n=1}^{N}$ by removing off-diagonal terms. However, this would also remove correlations from the active constraints, making the overall framework less expressive.

D. Adaptation with multiple actions

The ability to modulate the trajectory at multiple points from the same user command helps guarantee smoothness of the resulting generalized cylinder as well as a lower burden for the user. We thus further modify (3) to allow the application of P corrections. In practice, from the original $\hat{\boldsymbol{\xi}}$ computed with (4) a set of P desired modulations $\{\hat{\boldsymbol{\xi}}_i\}_{i=1}^{P}$ is computed, with the magnitude of each action decaying exponentially from the center $\hat{\boldsymbol{\xi}}_{\frac{P-1}{2}}$ and applied on $\hat{\boldsymbol{s}}_1, \ldots, \hat{\boldsymbol{s}}_P$. Note that applying multiple actions does not invalidate the decorelation strategy. Their effect on the original KMP can be computed cumulatively as[2]

$$\mathbb{E}[\boldsymbol{\xi}(\boldsymbol{s}^*)] = \boldsymbol{k}^*\boldsymbol{\Psi}\boldsymbol{\mu} + \sum_{i=1}^{P} \left[\hat{\boldsymbol{k}}_i^* - \boldsymbol{k}_i^*\boldsymbol{\Psi}\hat{\boldsymbol{K}}_i\right] \hat{\boldsymbol{\xi}}_i. \tag{6}$$

Removing the correlations between different DoFs (IV-C) is achieved by combining (5) and (6), yielding

$$\mathbb{E}[\boldsymbol{\xi}(\boldsymbol{s}^*)] = \boldsymbol{k}^*\boldsymbol{\Psi}\boldsymbol{\mu} + \sum_{i=1}^{P}\sum_{j=1}^{D} \bar{\boldsymbol{S}}_j \left[\hat{\boldsymbol{k}}_i^* - \boldsymbol{k}_i^*\boldsymbol{\Psi}\hat{\boldsymbol{K}}_i\right] \hat{\boldsymbol{\xi}}_{i,j}. \tag{7}$$

Equation (7) implements the modulations of a KMP at multiple points simultaneously while removing undesired effects between task space DoFs resulting from correlations in the training data. Note that the computational complexity[3] of (7) is $\mathcal{O}(pn^2)$. As a consequence, as $P \to N$, the complexity approaches that of the original via-point-based adaptation approach, $\mathcal{O}(n^3)$. This, however, is unlikely to occur in practice, as increasing P removes the locality of the modulation, which is rarely desirable.

V. Results

We conduct three experiments to validate our contribution, with 3D user commands: we first learn an SCT from demonstrated data, then adapt it to new environmental conditions (new target pose then new target object), and finally let users familiar with EDAN adapt an SCT on their own. Successful tasks executions on EDAN are shown for each experiment. Further results are shown in the accompanying video[4].

A. Learning active constraints for a picking skill

The goal of the first experiment was to pick up a cup from a table. To this end, we designed an SCT with two states: *approach* and *lift*. The transition from *approach* to *lift* was triggered by the user. For each state, five demonstrations were recorded by kinesthetic teaching. Dynamical Time Warping was then used to temporally align the demonstrated data. A KMP was fitted to the data using a GMM with two components to compute $\{\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n\}_{n=1}^{N}$, with $N = 100$, $\lambda_1 = 0.25$, $\lambda_2 = 0.25$, $\alpha = 0.75$, and a Matérn kernel [20] with $p = 2$ and length scale $l = 0.2$, chosen empirically.

[2] A term $\underline{K}^{-1}$ multiplies $\hat{\boldsymbol{\xi}}$ on the left in the original formulation, when multiple actions are applied. However, it correlates modulations at the action level and increases the complexity, with limited benefits for our current use case. We thus choose to treat here the individual actions independently.

[3] We neglect the effect of D as the projector does not depend on j.

[4] https://sites.google.com/view/learning-shared-control/

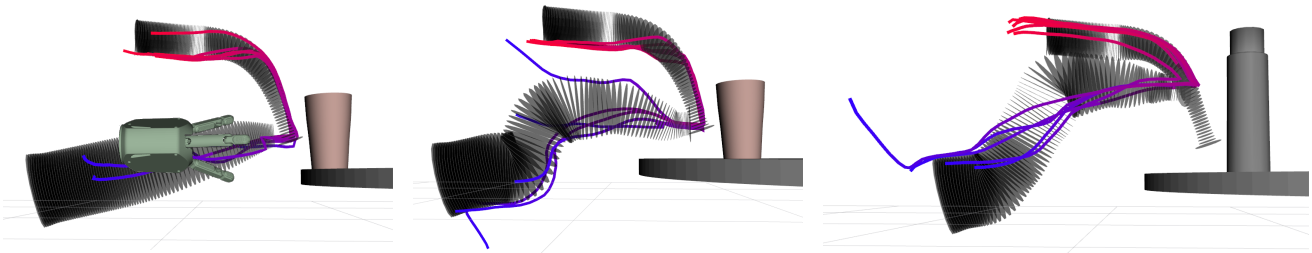

Fig. 3. Depiction of various SCTs executions on EDAN, with trajectories going from blue to red. **Left**: *Pick* SCT obtained by fitting a KMP to demonstrations. **Center**: *Pick* SCT adapted with the proposed method (in the *approach* state) such that the end-effector fingers do not hit the table. **Right**: A second adapted SCT, to pick up a taller object. The SCT implementation ensures that the end-effector is smoothly pulled to stay inside the active constraints.

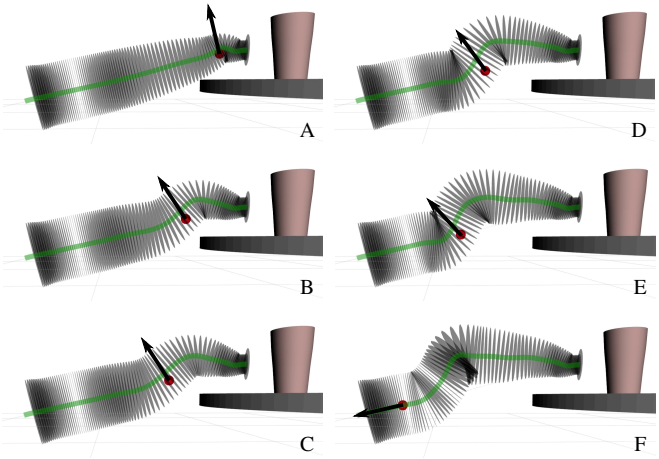

Fig. 4. Photo series of the iterative modulation of a learned approach constraint, to ensure the fingers of the end-effector will not collide with the table with the new cup placement. User commands are depicted with a black arrow.

Finally, for each state a generalized cylinder was derived from the KMP and used as active constraint, with a standard deviation of 5 to select the covariance ellipsoids. The end-effector orientation was extracted from the closest point of the mean trajectory. Five successful task executions by the first author with the learned SCT are shown in Fig. 3, *left*.

B. Adapting learned active constraints to new conditions

For the second experiment, the cup was placed closer to the center of the table. The (object-centric) trajectory had to be adapted, to not let the fingers collide with the table when using a approach trajectory with low height. Deforming the *approach* model in correction mode is shown in Fig. 4. In this simple scenario with a precise input device, $P = 1$ was chosen. The effect of increasing P can be visualized in the accompanying video. The red sphere indicates where the modulation is taking place, i. e. $\hat{\boldsymbol{s}}$, selected with $\boldsymbol{u}_{||}$. To make it easier to select $\hat{\boldsymbol{s}}$, if $\|\boldsymbol{u}_{||}\| > \|\boldsymbol{u}_{\perp}\|$, $\boldsymbol{u}_{\perp}$ is nullified, so that no modulation is applied if the user predominantly intend to move along the directrix. On exit of the correction mode, the model is updated, the SCT reloaded, then five pick tasks were performed successfully, see Fig 3, *center*. We then used a thermos as target object, which changed the terminal state as it required to be grasped higher, therefore requiring a second adaptation. Following the same procedure, five executions are shown in Fig. 3, *right*.

TABLE I

SKILL MODULATION BY DIFFERENT USERS.

KPI	**Participant**					
	Author	A	B	C	D	E
Modulation time (s)	**30**	71	87	79	40	62
Deformation %	**5**	10	34	18	10	20

C. Action decorrelation and computational complexity

Figure 5 shows the trajectories recorded for the *approach* state, and the resulting mean and variance from a fitted KMP model. Additionally, the impact of a null space action is shown, both on the original NS-KMP formulation [8] and our proposed approach. Figure 6 shows the computation time resulting from using null space actions or via-points to modulate the main trajectory. Two approaches of adding via-points are evaluated: either by increasing the set of points of the reference trajectory distribution, or by replacing points, keeping the set size constant.

D. Evaluating performance of new users

Finally, we evaluated how other team members would modulate the trajectory in the same setting as the first part of V-B: pick a cup placed closer to the center of the table. They had no experience with the proposed method, but are familiar with SCTs and joystick control. After being explained the SCT behavior, they tried the original SCT a few times (less than five), as in the first experiment. Then they practiced modulating the KMP model. At their convenience (in practice, within four minutes) they signaled they were ready to start to adjust the SCT to the new environment situation, and entered correction mode to do so. Finally they executed the adapted SCT on the robot, which was successful on first try for each user. The time it took each person to modulate the trajectory as well as the deformation applied on each trajectory, measured as the deformed trajectory length divided by the original trajectory length, are shown in Table I.

VI. DISCUSSION

A. Results

The results obtained in V-A and V-B show that SCTs – particularly their active constraints – can be learned using our proposed approach, and adapted using the same input device as for task completion. These results are particularly relevant for motor-impaired users who may not always have the ability to make physical corrections. In V-D multiple

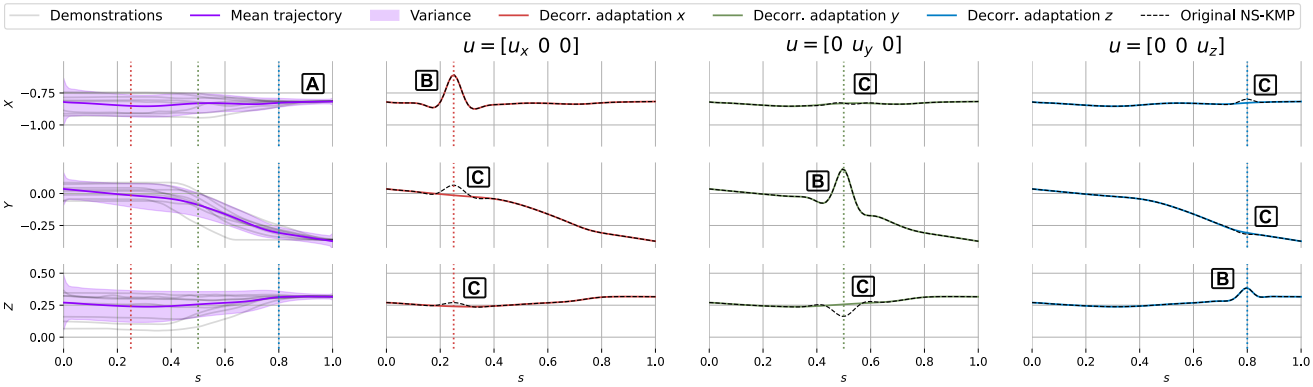

Fig. 5. **Null space action and decorrelation adaptation.** On the left are recorded trajectories to approach a cup, together with a fitted KMP model represented by its mean and variance. The variance decreases when getting closer to the target item A, which constrains the end-effector. In each subsequent column a null space action is applied along a single dimension. The impact of modulations, i. e. its amplitude scaled by the variance, can be seen at B. The effect of the proposed decorrelation adaptation can be seen at C, where a null space action in a single dimension has no effect in other dimensions. Notice the correlation effect occurring in the original NS-KMP formulation [8] (dashed black line) with actions on one DoF deforming other DoFs.

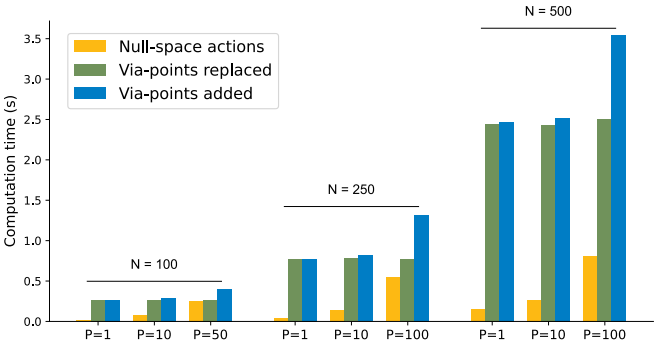

Fig. 6. Comparison of the computation time between null space action and via-points (mean computation time over 1000 runs). Via-points are specified to match the result of the applied null space action.

able-bodied users were able not only to complete the task but also to adapt the assistance provided by the robot to a new situation by modulating the demonstrated constraints. Table I shows that, even with no prior experience with the method, this adaptation could be done in realistic time frames and with applied deformations which are comparable to those of an expert user, here the leading author. These results suggest that the approach is intuitive and easy to use. In addition, Fig. 5 validates our action decorrelation strategy, showing that individual actions on one DoF do not affect other DoFs.

The results obtained in Fig. 6 with a non-optimized Python implementation confirm the trend discussed in IV-D with P increasing the computational time in our approach. Due to using a non-optimized implementation the complexity of our approach matches that of *via-points replaced* at $P = 0.5N$. Note, however, that this corresponds to modulating half of the constraint simultaneously. We observed empirically that $P < 0.2N$ was a sensible choice, ensuring a good trade-off between the locality of modulations and the number of commands applied by the user. We thus leveraged the benefits of the efficient modulation offered by (3). As shown in the accompanying video, this simple implementation gave us a low enough latency for model updates allowing intuitive modulations for a simple SCT. Additionally, both via-point adaptations entail the decrease in the predicted covariance profile which we deem undesirable in our setting.

B. Outlook and limitations

Regardless of how the adaptation is done – null space or via-point-based – we argue that KMP-based representations allow to learn and adapt constraints in SCTs, as they are object-centric and time independent, leading to identical shared control behaviors between trials, and provide local adaptation. The modular nature of SCTs also allow hybrid approaches where some states contain KMP-based constraints while others are hand-coded. Note that by design, in our experiments we discarded the possibility to modulate the KMP along the directrix, although the formulation allows to modulate in this direction as well if necessary.

Finally, the impact of $\boldsymbol{\Sigma}$ on the modulations, a consequence of the projector used in (3), provides more precise control of the more critical sections of the demonstrations. However, a set of trajectories with $\boldsymbol{\Sigma}$ varying too much (or with unintended too high/low variance) may also be unintuitive for the user. A formulation with an hyperparameter providing continuous adjustment between scaling due to $\boldsymbol{\Sigma}$ and no scaling on the null space action would be of interest. Also note that our approach modulates the active constraints via modulating the expectation of the KMP through (3). Our results show that this is sufficient to successfully adapt a picking skill in our scenario but a similar mechanism to modulate the covariance could also be investigated.

VII. Conclusion

We proposed an approach using probabilistic skill representations to learn and adapt shared control skills. We showed that using a skill representation with soft null space projectors [8] permits a computationally-efficient modulation of SCTs without decreasing motion freedom for users. Successful executions of a learned picking task, as well as the skill adaptation to new environment conditions, were shown on the assistive robot EDAN. The success achieved by new users in completing the task and adapting the skill suggests that the proposed approach is intuitive and easy to use.

In future work we will further investigate the intuitiveness and usability of the approach by considering new tasks and more participants with no robotics background.

REFERENCES

[1] B. Siciliano, O. Khatib, and T. Kröger, *Springer handbook of robotics*. Springer, 2008, vol. 200.

[2] J. Vogel, A. Hagengruber, M. Iskandar, G. Quere, U. Leipscher, S. Bustamante, A. Dietrich, H. Höppner, D. Leidner, and A. Albu-Schäffer, "Edan: An emg-controlled daily assistant to help people with physical disabilities," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4183–4190.

[3] J. Vogel, D. Leidner, A. Hagengruber, M. Panzirsch, B. Bauml, M. Denninger, U. Hillenbrand, L. Suchenwirth, P. Schmaus, M. Sewtz, *et al.*, "An ecosystem for heterogeneous robotic assistants in caregiving: Core functionalities and use cases," *IEEE Robotics & Automation Magazine*, vol. 28, no. 3, pp. 12–28, 2020.

[4] G. Quere, A. Hagengruber, M. Iskandar, S. Bustamante, D. Leidner, F. Stulp, and J. Vogel, "Shared control templates for assistive robotics," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1956–1962, 2020.

[5] S. A. Bowyer, B. L. Davies, and F. R. y Baena, "Active constraints/virtual fixtures: A survey," *IEEE Transactions on Robotics (T-RO)*, vol. 30, no. 1, pp. 138–157, 2013.

[6] Y. Huang, L. Rozo, J. Silvério, and D. G. Caldwell, "Kernelized movement primitives," *The International Journal of Robotics Research*, vol. 38, no. 7, pp. 833–852, 2019.

[7] S. R. Ahmadzadeh and S. Chernova, "Trajectory-based skill learning using generalized cylinders," *Frontiers in Robotics and AI*, vol. 5, p. 132, 2018.

[8] J. Silvério and Y. Huang, "A non-parametric skill representation with soft null space projectors for fast generalization," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2023, pp. 2988–2994.

[9] C. Celemin, R. Pérez-Dattari, E. Chisari, G. Franzese, L. de Souza Rosa, R. Prakash, Z. Ajanović, M. Ferraz, A. Valada, and J. Kober, "Interactive imitation learning in robotics: A survey," *Foundations and Trends® in Robotics*, vol. 10, no. 1-2, pp. 1–197, 2022.

[10] N. Mehr, R. Horowitz, and A. D. Dragan, "Inferring and assisting with constraints in shared autonomy," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 6689–6696.

[11] A. Broad, J. Arkin, N. Ratliff, T. Howard, and B. Argall, "Real-time natural language corrections for assistive robotic manipulators," *The International Journal of Robotics Research*, vol. 36, no. 5-7, pp. 684–698, 2017.

[12] M. Selvaggio, G. A. Fontanelli, F. Ficuciello, L. Villani, and B. Siciliano, "Passive virtual fixtures adaptation in minimally invasive robotic surgery," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3129–3136, 2018.

[13] G. Quere, S. Bustamante, A. Hagengruber, J. Vogel, F. Steinmetz, and F. Stulp, "Learning and interactive design of shared control templates," *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1887–1894, 2021.

[14] D. P. Losey, H. J. Jeon, M. Li, K. Srinivasan, A. Mandlekar, A. Garg, J. Bohg, and D. Sadigh, "Learning latent actions to control assistive robots," *Autonomous robots*, vol. 46, no. 1, pp. 115–147, 2022.

[15] C. Pérez-D'Arpino and J. Shah, "C-learn: Learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 4058–4065.

[16] I. Havoutis and S. Calinon, "Learning from demonstration for semi-autonomous teleoperation," *Autonomous Robots*, vol. 43, no. 3, pp. 713–726, March 2019.

[17] S. Iregui, J. De Schutter, and E. Aertbelien, "Reconfigurable constraint-based reactive framework for assistive robotics with adaptable levels of autonomy," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7397–7405, 2021.

[18] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 61, no. 3, pp. 611–622, 1999.

[19] G. Raiola, S. Sanchez Restrepo, P. Chevalier, P. Rodriguez-Ayerbe, X. Lamy, S. Tliba, and F. Stulp, "Co-manipulation with a Library of Virtual Guiding Fixtures," *Autonomous Robots*, pp. 1573–7527, 2018.

[20] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. Cambridge, MA, USA: MIT Press, 2006.

[21] D. P. Losey and M. K. O'Malley, "Trajectory deformations from physical human–robot interaction," *IEEE Trans. on Robotics*, vol. 34, no. 1, pp. 126–138, 2018.

[22] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intelligent Data Analysis*, vol. 11, 2007.

[23] P. P. Klein, "On the ellipsoid and plane intersection equation," *Applied Mathematics*, vol. 3, no. 11, pp. 1634–1640, 2012.