Proceedings Paper:
Huang, Y, Silvério, J, Rozo, L et al. (1 more author) (2018) Generalized Task-Parameterized Skill Learning. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). ICRA 2018, 21-25 May 2018, Brisbane, Australia. IEEE , pp. 5667-5474.

https://doi.org/10.1109/ICRA.2018.8461079

Generalized Task-Parameterized Skill Learning

Yanlong Huang, João Silvério, Leonel Rozo, and Darwin G. Caldwell

***Abstract*— Programming by demonstration has recently gained much attention due to its user-friendly and natural way to transfer human skills to robots. In order to facilitate the learning of multiple demonstrations and meanwhile generalize to new situations, a task-parameterized Gaussian mixture model (TP-GMM) has been recently developed. This model has achieved reliable performance in areas such as human-robot collaboration and dual-arm manipulation. However, the crucial task frames and associated parameters in this learning framework are often set by the human teacher, which renders three problems that have not been addressed yet: *(i)* task frames are treated equally, without considering their individual importance, *(ii)* task parameters are defined without taking into account additional task constraints, such as robot joint limits and motion smoothness, and *(iii)* a fixed number of task frames are pre-defined regardless of whether some of them may be redundant or even irrelevant for the task at hand. In this paper, we generalize the task-parameterized learning by addressing the aforementioned problems. Moreover, we provide a novel learning perspective which allows the robot to refine and adapt previously learned skills in a low dimensional space. Several examples are studied in both simulated and real robotic systems, showing the applicability of our approach.**

I. Introduction

As an intuitive and user-friendly way to endow a robot with skills from humans, Programming by Demonstration (PbD) has become appealing in the past few years [1]. The basic idea of PbD is to extract the important or consistent features from demonstrations and then adapt them to various situations, which is also referred to as generalization. In practice, a myriad of robot tasks are formulated as a regression problem, e.g., a mapping from sensory information to robot (motor) actions. However, typical regression approaches such as locally weighted regression (LWR) [2] or Gaussian process regression (GPR) [3] may suffer from limited extrapolation capabilities [4]. In order to adapt learned robot skills to a broader range of task instances, a multi-frame based probabilistic learning framework TP-GMM was proposed [4]. This approach exploits locally consistent features among demonstrations in different local coordinate systems instead of using a single global reference frame, and subsequently transfers local features to new task frames (which describe new task situations), yielding reliable performance for both interpolation and extrapolation.

However, the crucial task frames and associated parameters in TP-GMM are usually set according to the human knowledge about the task, which renders three main limitations: *(i)* task frames are treated equally without considering their individual importance. However, depending on human interpretation of tasks, the task frames influence may vary over time, which can be interpreted as the expertise or *confidence* that a specific frame has with respect to a portion of the task, which is overlooked in [4]; *(ii)* task parameters are defined regardless of additional task constraints, such as robot joint limits and motion smoothness. These new constraints demand the robot to adapt the learned task-parameterized skill according to additional requirements while performing successfully; *(iii)* a fixed number of task frames are predefined ignoring whether some of them are redundant or even irrelevant for the task at hand. These unnecessary frames will increase the computational burden and potentially degrade the overall performance of TP-GMM.

Besides the foregoing problems, it is worth mentioning that human demonstrations might not be optimal for the robot. Namely, the demonstrator may mainly focus on the task at hand while the robot capability is not fully exploited, which may lead to high energy movements, unnecessary large joint displacements, or high torque motion, among other problems. Also, due to the complicated structure or non-linearity exhibited in the demonstrated trajectories, it is non-trivial to optimize these trajectories effectively. We here propose to take advantage of the task-parametrized formulation of TP-GMM by optimizing task parameters instead of directly modifying the model parameters (i.e., GMM means and covariance matrices), while the latter is conventionally done [5]. A clear advantage of our approach is that task parameters lie in a lower dimensional space compared to that of trajectory model parameters.

In this paper, we first briefly introduce TP-GMM (Section II). Subsequently, we consider a variant of TP-GMM (Section III-A) so as to address the stated problems properly. Using this new formulation, we propose a confidence-weighted scheme to address problem *(i)* (Section III-B). In order to cope with problem *(ii)*, we formulate the optimization of task parameters as a reinforcement learning (RL) problem (Section IV-A), with the aim of enabling the robot to finish the task while satisfying additional task constraints. Also, we provide a dual perspective to show that the optimization of task parameters in a lower dimensional space is equivalent to that of model parameters in a higher dimensional space (Section IV-B). Furthermore, as a solution to problem *(iii)* we propose an iterative frame selection algorithm to exploit the most relevant task frames (Section V), where the task parameters optimization is used. Finally, we evaluate our approaches through several examples in Section VI and conclude this paper in Section VII. An overview of our main contributions is illustrated in Fig. 1.

All authors are with Department of Advanced Robotics, Istituto Italiano di Tecnologia, Via Morego 30, 16163 Genoa, Italy, firstname.lastname@iit.it

This work was supported by the Italian Minister of Defense.

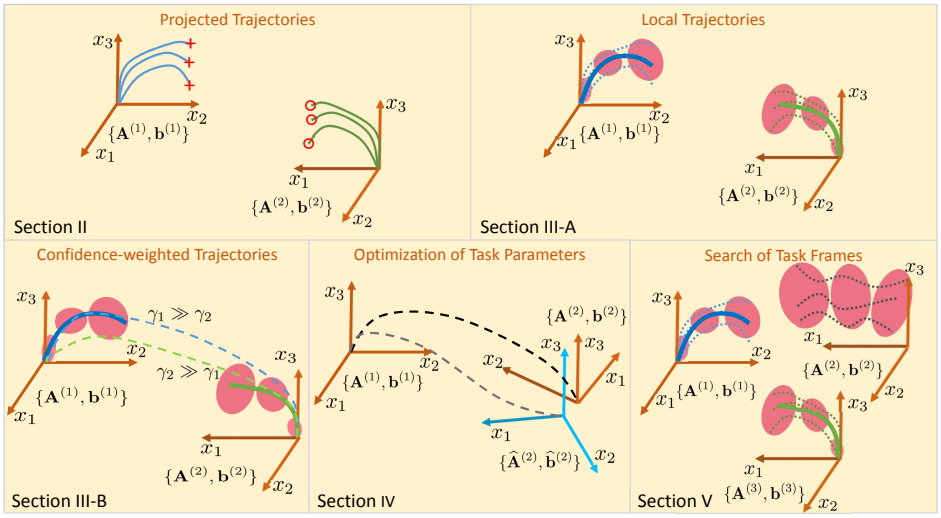

Fig. 1. Illustrations of task-parameterized movement learning. *Top-left* plot depicts projected trajectories in different task frames, where 'o' and '+' denote the start and end points of trajectories, respectively. *Top-right* plot shows trajectory encoding using GMM and trajectory retrieval using GMR, where the ellipses depict GMM components and the solid curves represent the mean trajectories retrieved by GMR. *Bottom-left* graph illustrates the trajectory generation using the confidence-weighted scheme. *Bottom-middle* graph presents the trajectory adaptation using the optimized task parameters. *Bottom-right* figure shows a case where the selection of task frames is important.

II. An Overview of Task-Parameterized Gaussian Mixture Model

In the context of imitation learning, one crucial ingredient is the consistent features underlying human demonstrations [4], [5], [6]. In order to facilitate the extraction of consistent features, TP-GMM has been exploited in numerous applications, e.g., human-robot collaborative transportation [7] as well as robot bimanual sweeping [8]. Often, a set of candidate task frames (e.g., frames at target objects [7] or robot end-effectors [8]) needs to be pre-defined for the implementation of TP-GMM.

Formally, let us consider P *task frames*, and refer to the rotation matrix $\mathbf{A}_t^{(j)}$ and translation vector $\mathbf{b}_t^{(j)}$ of each frame $\{j\}$ with respect to the global reference frame $\{O\}$ as the *task parameters*, where t denotes the time step and $j = 1, 2, \ldots, P$. We then project human demonstrations $\{\{\boldsymbol{\xi}_{t,m}\}_{t=1}^{N}\}_{m=1}^{M}$ into each frame separately and subsequently exploit the local features in different frames. Here, N and M respectively represent the time length of each demonstration and the number of demonstrations, while $\boldsymbol{\xi}_{t,m} \in \mathbb{R}^D$ represents a D-dimensional trajectory point. The projected trajectories in each frame $\{j\}$ are computed by (see [4] for details)

$$\boldsymbol{\xi}_{t,m}^{(j)} = (\mathbf{A}_t^{(j)})^{-1}(\boldsymbol{\xi}_{t,m} - \mathbf{b}_t^{(j)}). \tag{1}$$

If we consider the estimation of consistent features among the projected trajectories from a probabilistic perspective, GMM can be employed [4], [7], [8], [9], which has shown reliable modeling of joint distribution of trajectories. By using *Expectation Maximization* (EM) algorithm, GMM parameters $\{\pi_k, \{\boldsymbol{\mu}_k^{(j)}, \boldsymbol{\Sigma}_k^{(j)}\}_{j=1}^{P}\}_{k=1}^{K}$ in different frames can be estimated, where K represents the number of Gaussian components, π_k, $\boldsymbol{\mu}_k^{(j)}$ and $\boldsymbol{\Sigma}_k^{(j)}$ respectively denote mixture coefficients, Gaussian centers and covariance matrices in each frame $\{j\}$.

By using affine transformations and product of Gaussians, new GMM components $\{\pi_k, \boldsymbol{\mu}_{k,t}, \boldsymbol{\Sigma}_{k,t}\}_{k=1}^{K}$ at time t in the global frame $\{O\}$ can be computed as

$$\mathcal{N}(\boldsymbol{\mu}_{k,t}, \boldsymbol{\Sigma}_{k,t}) \propto \prod_{j=1}^{P} \mathcal{N}\Big(\mathbf{A}_t^{(j)}\boldsymbol{\mu}_k^{(j)} + \mathbf{b}_t^{(j)}, \mathbf{A}_t^{(j)}\boldsymbol{\Sigma}_k^{(j)}(\mathbf{A}_t^{(j)})^T\Big), \tag{2}$$

which yields a distribution $\boldsymbol{\xi}_t \sim \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{\mu}_{k,t}, \boldsymbol{\Sigma}_{k,t})$ in the frame $\{O\}$. Furthermore, we can decompose $\boldsymbol{\xi}$ into input $\boldsymbol{\xi}_{\mathcal{I}}$ and output $\boldsymbol{\xi}_{\mathcal{O}}$, and subsequently, generate a trajectory in the global frame $\{O\}$ as $\boldsymbol{\xi}_{t,\mathcal{O}}|\boldsymbol{\xi}_{t,\mathcal{I}} \sim \mathcal{N}(\boldsymbol{\mu}_{t,\mathcal{O}}, \boldsymbol{\Sigma}_{t,\mathcal{O}})$ by using Gaussian mixture regression (GMR) [4], [10]. To name an example, if we consider $\boldsymbol{\xi}_{\mathcal{I}}$ and $\boldsymbol{\xi}_{\mathcal{O}}$ as time and a 3-D trajectory point, respectively, then a sequence of trajectory points in frame $\{O\}$ at different time steps can be generated. Note that the input is not limited to be time, other inputs can be possible depending on the task characteristics.

III. Confidence-Weighted Task-Parameterized Movement Learning

In order to formulate the confidence assignments to task frames, optimization of task parameters as well as frame selection, we first introduce a variant of TP-GMM in Section III-A. Note that intuitive insights on this variant have been studied for two frames [11] and multiple frames [12], while here we aim to provide a mathematical proof. Subsequently, we propose a novel confidence-weighted scheme (described in Section III-B) which, for example, allows the demonstrator to include information about his/her confidence regarding the relevance/influence of each task frame with respect to the task that is being learned.

A. Task-parameterized Movement Trajectories

Assuming that we can access to the local GMM models in different task frames, the local trajectory distribution in each frame $\{j\}$ can be represented as $\boldsymbol{\xi}^{(j)} \sim \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{\mu}_k^{(j)}, \boldsymbol{\Sigma}_k^{(j)})$. By decomposing $\boldsymbol{\xi}^{(j)}$ as the input $\boldsymbol{\xi}_{\mathcal{I}}^{(j)}$ and the output $\boldsymbol{\xi}_{\mathcal{O}}^{(j)}$, we can generate a local trajectory in frame $\{j\}$ as $\boldsymbol{\xi}_{t,\mathcal{O}}^{(j)} | \boldsymbol{\xi}_{t,\mathcal{I}}^{(j)} \sim \mathcal{N}(\boldsymbol{\mu}_{t,\mathcal{O}}^{(j)}, \boldsymbol{\Sigma}_{t,\mathcal{O}}^{(j)})$ using GMR. The global trajectory can be viewed as a trade-off among all local trajectories. Formally, the global trajectory $\boldsymbol{\xi}_{t,\mathcal{O}}$ can be estimated by maximizing $\prod_{j=1}^{P} \mathcal{P}\left(\boldsymbol{\xi}_{t,\mathcal{O}} | \mathbf{A}_{t,\mathcal{O}}^{(j)} \boldsymbol{\mu}_{t,\mathcal{O}}^{(j)} + \mathbf{b}_{t,\mathcal{O}}^{(j)}, \mathbf{A}_{t,\mathcal{O}}^{(j)} \boldsymbol{\Sigma}_{t,\mathcal{O}}^{(j)} (\mathbf{A}_{t,\mathcal{O}}^{(j)})^T\right)$. Through the logarithmic transformation, this objective can be solved by minimizing the cost function

$$J(\boldsymbol{\xi}_{t,\mathcal{O}}) = \sum_{j=1}^{P} (\boldsymbol{\xi}_{t,\mathcal{O}} - \mathbf{A}_{t,\mathcal{O}}^{(j)} \boldsymbol{\mu}_{t,\mathcal{O}}^{(j)} - \mathbf{b}_{t,\mathcal{O}}^{(j)})^T (\mathbf{A}_{t,\mathcal{O}}^{(j)} \boldsymbol{\Sigma}_{t,\mathcal{O}}^{(j)} (\mathbf{A}_{t,\mathcal{O}}^{(j)})^T)^{-1} (\boldsymbol{\xi}_{t,\mathcal{O}} - \mathbf{A}_{t,\mathcal{O}}^{(j)} \boldsymbol{\mu}_{t,\mathcal{O}}^{(j)} - \mathbf{b}_{t,\mathcal{O}}^{(j)}). \tag{3}$$

By calculating derivatives of (3) with respect to $\boldsymbol{\xi}_{t,\mathcal{O}}$, the optimal solution can be derived, which is equivalent to the product of Gaussians, i.e.,

$$\boldsymbol{\xi}_{t,\mathcal{O}} \sim \prod_{j=1}^{P} \mathcal{N}\left(\mathbf{A}_{t,\mathcal{O}}^{(j)} \boldsymbol{\mu}_{t,\mathcal{O}}^{(j)} + \mathbf{b}_{t,\mathcal{O}}^{(j)}, \mathbf{A}_{t,\mathcal{O}}^{(j)} \boldsymbol{\Sigma}_{t,\mathcal{O}}^{(j)} (\mathbf{A}_{t,\mathcal{O}}^{(j)})^T\right), \tag{4}$$

where $\mathbf{A}_{t,\mathcal{O}}^{(j)}$ and $\mathbf{b}_{t,\mathcal{O}}^{(j)}$ respectively correspond to the output blocks of $\mathbf{A}_t^{(j)} = blockdiag(\mathbf{A}_{t,\mathcal{I}}^{(j)}, \mathbf{A}_{t,\mathcal{O}}^{(j)})$ and $\mathbf{b}_t^{(j)} = [(\mathbf{b}_{t,\mathcal{I}}^{(j)})^T \ (\mathbf{b}_{t,\mathcal{O}}^{(j)})^T]^T$. Note that the input blocks $\mathbf{A}_{t,\mathcal{I}}^{(j)}$ and $\mathbf{b}_{t,\mathcal{I}}^{(j)}$ are used to retrieve the local desired inputs $\boldsymbol{\xi}_{t,\mathcal{I}}^{(j)}$ projected into the different task frames, which act as the conditional inputs for the generation of local trajectories. An illustration of trajectory encoding via GMM and trajectory retrieval via GMR is provided in Fig. 1. It can be observed from Fig. 1(*top-right*) that, the first Gaussian component in frame $\{1\}$ and the third component in frame $\{2\}$ (counting from left to right) have the smallest covariances, implying that trajectory segments encapsulated by these components are highly consistent across demonstrations, and therefore represent an important feature of the movements.

B. Confidence-weighted Task-parameterized Movement Learning

Among previous works on task-parameterized learning [4], [7], [8], task frames and associated parameters $\{\mathbf{A}_t^{(j)}, \mathbf{b}_t^{(j)}\}$ were defined beforehand. Moreover, task frames were assigned with equal priorities. However, it may happen that, for some specific task frames, their influences are expected to be larger than the rest of frames, and hence it is desired to introduce human confidence about task frames. On the basis of (4), the human prior information can be naturally incorporated into task frames. Assuming that the confidences of different task frames are known, let us denote them as $c_{t,j} \in (0, 1)$. We then formulate the original objective of the variant of TP-GMM as $\prod_{j=1}^{P} \mathcal{P}\left(\boldsymbol{\xi}_{t,\mathcal{O}} | \mathbf{A}_{t,\mathcal{O}}^{(j)} \boldsymbol{\mu}_{t,\mathcal{O}}^{(j)} + \mathbf{b}_{t,\mathcal{O}}^{(j)}, \mathbf{A}_{t,\mathcal{O}}^{(j)} \boldsymbol{\Sigma}_{t,\mathcal{O}}^{(j)} (\mathbf{A}_{t,\mathcal{O}}^{(j)})^T\right)^{c_{t,j}}$. Here, $c_{t,j}$ can be interpreted as a measurement of the contribution of each local conditional Gaussian distribution to the product operation. Similar to the derivation of (4), the optimal estimation of $\boldsymbol{\xi}_{t,\mathcal{O}}$ can be determined by

$$\boldsymbol{\xi}_{t,\mathcal{O}} \sim \prod_{j=1}^{P} \mathcal{N}\left(\mathbf{A}_{t,\mathcal{O}}^{(j)} \boldsymbol{\mu}_{t,\mathcal{O}}^{(j)} + \mathbf{b}_{t,\mathcal{O}}^{(j)}, \mathbf{A}_{t,\mathcal{O}}^{(j)} (\boldsymbol{\Sigma}_{t,\mathcal{O}}^{(j)} / c_{t,j}) (\mathbf{A}_{t,\mathcal{O}}^{(j)})^T\right). \tag{5}$$

The above result has an intuitive interpretation: if the frame $\{j\}$ has a higher (lower) confidence $c_{t,j}$ at time t, its contribution to the Gaussian product is higher (lower) due to a smaller (larger) covariance , i.e., $\boldsymbol{\Sigma}_{t,\mathcal{O}}^{(j)} / c_{t,j}$. Figure 1(*bottom-left*) depicts an example of applying confidence-weighted scheme, where the resulting trajectory favors local trajectory in the task frame that is assigned with a higher confidence.

IV. Optimization of Task-parameterized Movement Trajectories

In this section we address the question: how can good task parameters be selected? For instance, for applications with flexible task parameters, i.e., different values of task parameters allow for finishing the same task, which configuration of parameters is better? We tackle this problem by optimizing task parameters from a reinforcement learning perspective (Section IV-A), and subsequently, we provide a dual perspective on this optimization, so that a connection between our approach and the standard optimization of GMM components is built (Section IV-B).

A. Reinforcement Learning of Task Parameters

Considering that task parameters $\{\mathbf{A}_t^{(j)}, \mathbf{b}_t^{(j)}\}$ describe different task frames (and therefore, different task situations), a straightforward way to refine them is by applying rotation and translation operations to their pre-defined values. Since the input blocks in task parameters are often uncontrollable (e.g., a time sequence input), we only discuss the learning of output blocks, i.e., $\{\mathbf{A}_{t,\mathcal{O}}^{(j)}, \mathbf{b}_{t,\mathcal{O}}^{(j)}\}$. Formally, let us define new rotation matrices and translational vectors as $\{\mathbf{R}_t^{(j)}\}_{j=1}^{P}$ and $\{\mathbf{d}_t^{(j)}\}_{j=1}^{P}$, respectively. Then, for an arbitrary local trajectory point $\boldsymbol{\xi}_{t,\mathcal{O}}^{(j)}$ in the frame $\{j\}$, after new rotational and translational operations are performed, we can prove that its representation in the reference frame $\{O\}$ becomes

$$\hat{\boldsymbol{\xi}}_{t,\mathcal{O}}^{(j)} = \underbrace{\mathbf{A}_{t,\mathcal{O}}^{(j)} \mathbf{R}_t^{(j)}}_{\widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)}} \boldsymbol{\xi}_{t,\mathcal{O}}^{(j)} + \underbrace{\mathbf{A}_{t,\mathcal{O}}^{(j)} \mathbf{d}_t^{(j)} + \mathbf{b}_{t,\mathcal{O}}^{(j)}}_{\widehat{\mathbf{b}}_{t,\mathcal{O}}^{(j)}}. \tag{6}$$

Accordingly, new task parameters $\{\widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)}, \widehat{\mathbf{b}}_{t,\mathcal{O}}^{(j)}\}$ of the frame $\{j\}$ are determined, which can be later used to replace initial task parameters $\{\mathbf{A}_{t,\mathcal{O}}^{(j)}, \mathbf{b}_{t,\mathcal{O}}^{(j)}\}$ and generate a new trajectory sequence in the frame $\{O\}$ via (4). With the affine transformation in (6), we actually learn task parameters by finding the optimal rotation matrices $\mathbf{R}_t^{(j)}$ and translational vectors $\mathbf{d}_t^{(j)}$.

We here consider that the rotation matrix represents sequential rotations about (x, y, z) axes with angles (α, β, γ),

and therefore the determination of $\{\mathbf{R}_t^{(j)}\}_{j=1}^P$ is equivalent to that of $\{\alpha_t^{(j)}, \beta_t^{(j)}, \gamma_t^{(j)}\}_{j=1}^P$. Furthermore, let us denote $\mathbf{a}_t^{(j)} = [\alpha_t^{(j)} \ \beta_t^{(j)} \ \gamma_t^{(j)} \ \mathbf{d}_t^{(j)T}]$ and $\mathbf{a}_t = [\mathbf{a}_t^{(1)} \ \mathbf{a}_t^{(2)} \ \cdots \ \mathbf{a}_t^{(P)}]^T$. In order to formulate the learning of $\mathbf{a}_t$ into a RL problem, we represent $\mathbf{a}_t$ as a parametric policy, i.e.,

$$\mathbf{a}_t = \mathbf{\Phi}_t(\boldsymbol{\theta} + \boldsymbol{\epsilon}), \tag{7}$$

where $\mathbf{\Phi}_t$ and $\boldsymbol{\epsilon}$ represent basis functions and stochastic exploration noise, respectively, and $\boldsymbol{\theta}$ denotes the policy parameters to be learned. By optimizing $\boldsymbol{\theta}$ with respect to additional constraints (i.e., task-dependent cost functions), the optimal parameters $\mathbf{a}_t$ can be found, which are subsequently used to retrieve new task parameters based on (6). Since we focus on optimizing the task parameters associated with task frames, the rotation matrix (defined by rotation angles) is an intuitive way to modify frames. Also, here we focus on learning positions rather than orientations, and thus this rotation operation suffices for our optimization problem.

For the typical policy search problem (7), many algorithms have been proven effective. Here, we take policy improvement with path integrals (PI^2) [13], [14] as an example to illustrate the reinforcement learning of task parameters. Let us denote the exploration noise at time step $i \in \{1, 2, \ldots, N\}$ during the roll-out (i.e., episode) $h \in \{1, 2, \ldots, H\}$ as $\boldsymbol{\epsilon}_{i,h}$, where N is the time length of a roll-out and H is the number of roll-outs. As suggested in [15], we apply a constant exploration noise $\boldsymbol{\epsilon}_h$ during the h-th roll-out (i.e., $\boldsymbol{\epsilon}_h = \boldsymbol{\epsilon}_{i,h}, \forall i \in \{1, 2, \ldots, N\}$) and update the policy parameters using every H roll-outs. In each roll-out, we can first calculate $\mathbf{a}_t$ using (7). Subsequently, we can retrieve new task parameters $\{\widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)}, \widehat{\mathbf{b}}_{t,\mathcal{O}}^{(j)}\}_{j=1}^P$ using (6). By plugging new task parameters and local trajectories in different frames into (4), an updated trajectory in the reference frame can be generated. Moreover, on the basis of the cost function (which is usually pre-defined depending on the specific task and additional constrains), we can compute the cumulative cost value S_h for each roll-out. Thus, given the cumulative costs $\{S_h\}_{h=1}^H$ in H roll-outs, the policy parameters are updated as follows

$$\boldsymbol{\theta} := \boldsymbol{\theta} + \sum_{h=1}^{H} w_h \boldsymbol{\epsilon}_h, \tag{8}$$

with $w_h = \frac{e^{-\kappa S_h}}{\sum_{h=1}^H e^{-\kappa S_h}}$ and $\kappa > 0$. We can continuously perform explorations and update $\boldsymbol{\theta}$ every H roll-outs until $\boldsymbol{\theta}$ converges or the cumulative cost is below a certain value. The complete learning procedure is illustrated in *Algorithm* 1. Figure 1(*bottom-middle*) shows trajectory adaptation using optimized task parameters. As it can be seen, the final trajectory is modulated by using new task parameters.

B. Dual Perspective of Optimizing Task Parameters

In this section, we provide a dual perspective to interpret the optimization of task parameters. To do so, let us first recall the main result (2) in TP-GMM. Note that in (2), there exists an affine transformation of the GMM component $\{\boldsymbol{\mu}_k^{(j)}, \mathbf{\Sigma}_k^{(j)}\}$ through the task parameters $\{\mathbf{A}_t^{(j)}, \mathbf{b}_t^{(j)}\}$. If we write the block-decomposition of $\boldsymbol{\mu}_k^{(j)} = \begin{bmatrix} \boldsymbol{\mu}_{k,\mathcal{I}}^{(j)} \\ \boldsymbol{\mu}_{k,\mathcal{O}}^{(j)} \end{bmatrix}$, $\mathbf{\Sigma}_k^{(j)} = \begin{bmatrix} \mathbf{\Sigma}_{k,\mathcal{I}\mathcal{I}}^{(j)} & \mathbf{\Sigma}_{k,\mathcal{I}\mathcal{O}}^{(j)} \\ \mathbf{\Sigma}_{k,\mathcal{O}\mathcal{I}}^{(j)} & \mathbf{\Sigma}_{k,\mathcal{O}\mathcal{O}}^{(j)} \end{bmatrix}$ and substitute the optimized task parameters $\{\widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)}, \widehat{\mathbf{b}}_{t,\mathcal{O}}^{(j)}\}$ into (2), new mean and covariance can be derived as

$$\begin{aligned} \widehat{\boldsymbol{\mu}}_{t,k}^{(j)} &= \begin{bmatrix} \mathbf{A}_{t,\mathcal{I}}^{(j)} & \mathbf{0} \\ \mathbf{0} & \widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu}_{k,\mathcal{I}}^{(j)} \\ \boldsymbol{\mu}_{k,\mathcal{O}}^{(j)} \end{bmatrix} + \begin{bmatrix} \mathbf{b}_{t,\mathcal{I}}^{(j)} \\ \widehat{\mathbf{b}}_{t,\mathcal{O}}^{(j)} \end{bmatrix} \\ \widehat{\mathbf{\Sigma}}_{t,k}^{(j)} &= \begin{bmatrix} \mathbf{A}_{t,\mathcal{I}}^{(j)} & \mathbf{0} \\ \mathbf{0} & \widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)} \end{bmatrix} \begin{bmatrix} \mathbf{\Sigma}_{k,\mathcal{I}\mathcal{I}}^{(j)} & \mathbf{\Sigma}_{k,\mathcal{I}\mathcal{O}}^{(j)} \\ \mathbf{\Sigma}_{k,\mathcal{O}\mathcal{I}}^{(j)} & \mathbf{\Sigma}_{k,\mathcal{O}\mathcal{O}}^{(j)} \end{bmatrix} \begin{bmatrix} \mathbf{A}_{t,\mathcal{I}}^{(j)} & \mathbf{0} \\ \mathbf{0} & \widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)} \end{bmatrix}^T. \end{aligned} \tag{9}$$

This new mean and covariance can also be seen as being equivalent to a new local model $\{\widehat{\boldsymbol{\mu}}_k^{(j)}, \widehat{\mathbf{\Sigma}}_k^{(j)}\}$, rotated and translated by the old parameters $\{\mathbf{A}_t^{(j)}, \mathbf{b}_t^{(j)}\}$, resulting in

$$\begin{aligned} \widehat{\boldsymbol{\mu}}_{t,k}^{(j)} &= \begin{bmatrix} \mathbf{A}_{t,\mathcal{I}}^{(j)} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{t,\mathcal{O}}^{(j)} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\mu}}_{k,\mathcal{I}}^{(j)} \\ \widehat{\boldsymbol{\mu}}_{k,\mathcal{O}}^{(j)} \end{bmatrix} + \begin{bmatrix} \mathbf{b}_{t,\mathcal{I}}^{(j)} \\ \mathbf{b}_{t,\mathcal{O}}^{(j)} \end{bmatrix} \\ \widehat{\mathbf{\Sigma}}_{t,k}^{(j)} &= \begin{bmatrix} \mathbf{A}_{t,\mathcal{I}}^{(j)} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{t,\mathcal{O}}^{(j)} \end{bmatrix} \begin{bmatrix} \widehat{\mathbf{\Sigma}}_{k,\mathcal{I}\mathcal{I}}^{(j)} & \widehat{\mathbf{\Sigma}}_{k,\mathcal{I}\mathcal{O}}^{(j)} \\ \widehat{\mathbf{\Sigma}}_{k,\mathcal{O}\mathcal{I}}^{(j)} & \widehat{\mathbf{\Sigma}}_{k,\mathcal{O}\mathcal{O}}^{(j)} \end{bmatrix} \begin{bmatrix} \mathbf{A}_{t,\mathcal{I}}^{(j)} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{t,\mathcal{O}}^{(j)} \end{bmatrix}^T. \end{aligned} \tag{10}$$

By rewriting both (9) and (10) in their expanded forms, we have that

$$\begin{aligned} &\widehat{\boldsymbol{\mu}}_{k,\mathcal{I}}^{(j)} = \boldsymbol{\mu}_{k,\mathcal{I}}^{(j)} \\ &\widehat{\boldsymbol{\mu}}_{k,\mathcal{O}}^{(j)} = \left(\mathbf{A}_{t,\mathcal{O}}^{(j)}\right)^{-1} \widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)} \boldsymbol{\mu}_{k,\mathcal{O}}^{(j)} + \left(\mathbf{A}_{t,\mathcal{O}}^{(j)}\right)^{-1} \left(\widehat{\mathbf{b}}_{t,\mathcal{O}}^{(j)} - \mathbf{b}_{t,\mathcal{O}}^{(j)}\right) \\ &\widehat{\mathbf{\Sigma}}_{k,\mathcal{I}\mathcal{I}}^{(j)} = \mathbf{\Sigma}_{k,\mathcal{I}\mathcal{I}}^{(j)} \\ &\widehat{\mathbf{\Sigma}}_{k,\mathcal{O}\mathcal{I}}^{(j)} = \left(\mathbf{A}_{t,\mathcal{O}}^{(j)}\right)^{-1} \widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)} \mathbf{\Sigma}_{k,\mathcal{O}\mathcal{I}}^{(j)} \\ &\widehat{\mathbf{\Sigma}}_{k,\mathcal{I}\mathcal{O}}^{(j)} = \left(\widehat{\mathbf{\Sigma}}_{k,\mathcal{O}\mathcal{I}}^{(j)}\right)^T \\ &\widehat{\mathbf{\Sigma}}_{k,\mathcal{O}\mathcal{O}}^{(j)} = \left(\mathbf{A}_{t,\mathcal{O}}^{(j)}\right)^{-1} \widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)} \mathbf{\Sigma}_{k,\mathcal{O}\mathcal{O}}^{(j)} \left(\left(\mathbf{A}_{t,\mathcal{O}}^{(j)}\right)^{-1} \widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)}\right)^T. \end{aligned} \tag{11}$$

Thus, the optimization of task parameters (i.e., transform $\{\mathbf{A}_{t,\mathcal{O}}^{(j)}, \mathbf{b}_{t,\mathcal{O}}^{(j)}\}$ into $\{\widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)}, \widehat{\mathbf{b}}_{t,\mathcal{O}}^{(j)}\}$) is equivalent to the optimization of GMM components (i.e., transform $\{\boldsymbol{\mu}_k^{(j)}, \mathbf{\Sigma}_k^{(j)}\}$ into $\{\widehat{\boldsymbol{\mu}}_k^{(j)}, \widehat{\mathbf{\Sigma}}_k^{(j)}\}$). Note that rotation angles and translational vectors in (6) are learned to optimize task parameters. In contrast to classic approaches where GMM parameters (i.e., means, covariances and mixture coefficients) are updated [5], learning task parameters renders a lower dimensional optimization, which may speed up the learning process. More importantly, the optimization of task parameters is independent from the model parameters, and thus local consistent features from demonstrations are still maintained, which might be highly desirable for imitation learning.

V. Forward Search of Task Frames

Within the task-parameterized learning framework, the number of task frames are usually fixed and pre-determined [4], [7], [8]. A natural question concerning the number of

Algorithm 1 Optimization of task-parameterized movements

Initialization

1: Define a global reference frame $\{\mathrm{O}\}$ and initial candidate task frames $\{j\}_{j=1}^{P}$.

2: Collect demonstrations $\{\{\boldsymbol{\xi}_{t,m}\}_{t=1}^{N}\}_{m=1}^{M}$.

Phase 1: learn from demonstrations

1: Project demonstrations into each frame via (1) separately.

2: Fit GMM to projected trajectories in each frame using EM and generate local trajectories from conditional probabilities $\mathcal{P}(\boldsymbol{\xi}_{t,\mathcal{O}}^{(j)}|\boldsymbol{\xi}_{t,\mathcal{I}}^{(j)})$ in different frames using GMR.

Phase 2: generalization with optimized task parameters

1: Define new task parameters $\{\{\mathbf{A}_t^{(j)}, \mathbf{b}_t^{(j)}\}_{t=1}^{N}\}_{j=1}^{P}$ depending on the new task instance.

2: Optimize $\{\{\mathbf{A}_t^{(j)}, \mathbf{b}_t^{(j)}\}_{t=1}^{N}\}_{j=1}^{P}$ using (8) to minimize the cost function defined based on task requirements.

3: Use optimized task parameters $\{\{\widehat{\mathbf{A}}_{t,\mathcal{O}}^{(j)}, \widehat{\mathbf{b}}_{t,\mathcal{O}}^{(j)}\}_{t=1}^{N}\}_{j=1}^{P}$, combined with local trajectories in different task frames, to estimate $\boldsymbol{\xi}_{t,\mathcal{O}}$ in $\{O\}$ via (4).

task frames arises: can we change the number of task frames? More specifically, how can we determine the number of task frames? For instance, in a robot task with many candidate frames, redundant or irrelevant frames might exist, thus it is reasonable to remove these less important task frames so as to alleviate their undesired influences. As shown in Fig. 1(*bottom-right*), the task frame $\{2\}$ should be removed since this frame fails to encapsulate any consistent features from demonstrations. Even though this problem may have a significant impact on the robot performance, it has not been addressed in the previous works.

In analogy to the classical *forward search* used for selecting high dimensional features, we propose an iterative learning scheme to select the most-relevant task frames with respect to additional task constraints (which can be formulated as a cost function). We first consider the trajectory generation using a single frame. Through separate optimization of task parameters of each candidate frame via *Algorithm* 1, we can evaluate the influence of each frame based on their corresponding cost values. Note that the important frames influence the task significantly, and thus their corresponding cost values should decrease rapidly over the learning iterations. In other words the lower the cost, the higher the importance of the frame. With this insight, we can find the best frame in terms of cost values. Subsequently, we consider the trajectory generation using two frames, i.e., the best frame and one from the remaining ones. By evaluating the combination of the best frame and each of the other frames, the optimal two-frames set can be determined. Similarly, we can find the optimal frame set with more frames until the number of task frames reaches the upper limit. Note that the frame selection scheme depends on the definition of the cost function, which is closely related to the task requirements.

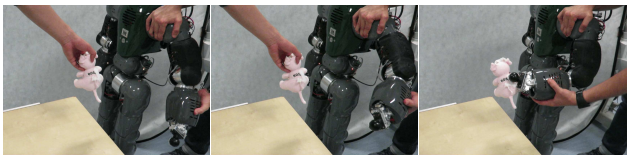

Fig. 2. Kinesthetic teaching of a reaching skill on the COMAN robot.

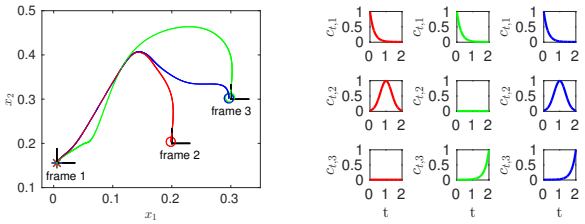

Fig. 3. Trajectories (*left* graph) generated by assigning different confidences (*right* plot) to task frames. The first, second and third columns of the *right* graph are associated with trajectories depicted by red, green and blue curves in the *left* graph, respectively. The start and end point of each trajectory are denoted by '$*$' and '$\circ$', respectively.

VI. Evaluations

In this section, we evaluate our proposed methods using several examples on the simulated/real COMAN robot [16]: *(i)* we consider the confidence-weighted scheme with different sets of frame confidences (Section VI-A) in the simulated robot; *(ii)* we apply the task frame optimization to a simulated task of reaching a pole (Section VI-B.1), a simulated and real tasks of reaching a point (Section VI-B.2) and a real reaching task with obstacle avoidance (Section VI-B.3); *(iii)* we implement a simulated pick-and-place task to show the frame selection procedure (Section VI-C). Since the tasks are all learned in the robot task space, we use the Jacobian matrix $\mathbf{J}$ of the robot end-effector to control the joint movement, i.e., $\widehat{\mathbf{q}}_{t+1} = \mathbf{q}_t + \mathbf{J}(\mathbf{q}_t)^{\dagger}(\widehat{\mathbf{p}}_{t+1} - \mathbf{p}_t)$ with $\mathbf{J}^{\dagger} = \mathbf{J}^T(\mathbf{J}\mathbf{J}^T)^{-1}$, $\mathbf{q}_t$ and $\mathbf{p}_t$ respectively represent the joint and Cartesian positions at time t, $\widehat{\mathbf{q}}_{t+1}$ and $\widehat{\mathbf{p}}_{t+1}$ represent the desired positions at time $t+1$.

A. Confidence-weighted Scheme

We collected 10 reaching trajectories with data-points represented by $\boldsymbol{\xi}_t = [t \quad \mathbf{p}^T]^T$ in the robot base frame $\{O\}$ using kinesthetic teaching on the COMAN's left arm (as shown in Fig. 2), lasting around $2s$ each. Assuming that the object orientation does not influence the reaching task, we define two initial frames located respectively at the start and end points of each demonstration. Through projecting demonstrations into these frames separately, we train a 4-states GMM to extract local consistency among projected trajectories in each frame, which is after used to retrieve local trajectories using GMR. Then, we consider the generalization of local trajectories to new task frames. Note that the robot arm starts from the same position, we therefore use the same start frame, i.e., frame $\{1\}$ in Fig. 3

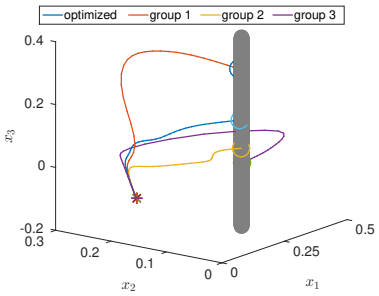

Fig. 4. Trajectories generated by using different task parameters from Table I in the task of reaching a static pole.

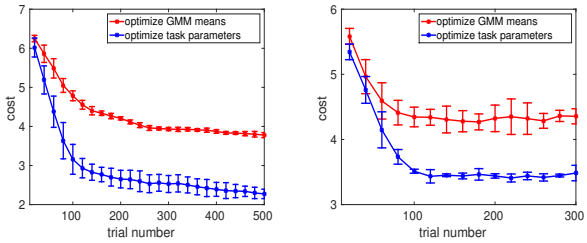

Fig. 5. These graphs show cost values of optimizing GMM means and task parameters in different reaching tasks, where the *left* and *right* figures correspond to the target object located at $[0.3\ \ 0.3\ \ 0.2]^T$ and $[-0.3\ \ 0.2\ \ 0.2]^T$ in frame $\{O\}$, respectively. Error-bars represent means and standard deviations of cost values.

(*left* graph), described by task parameters $\mathbf{A}^{(1)} = \mathbf{I}_{4\times4}$ and $\mathbf{b}^{(1)} = [0\ 0.005\ 0.156\ \ -0.050]^T$.

In order to illustrate the impact of frame confidences, we consider two new targets which are respectively located at $[0.2\ \ 0.2\ \ 0.2]^T$ and $[0.3\ \ 0.3\ \ 0.2]^T$, and thus we define two corresponding target frames $\{2\}$ and $\{3\}$ represented by task parameters $\mathbf{A}^{(2)} = \mathbf{A}^{(3)} = \mathbf{I}_{4\times4}$, $\mathbf{b}^{(2)} = [0\ 0.2\ 0.2\ 0.2]^T$, $\mathbf{b}^{(3)} = [0\ 0.3\ 0.3\ 0.2]^T$. Three different groups of frame confidences are evaluated, where each group corresponds to a column in Fig. 3 (*right* graph). The final trajectories in $\{O\}$ are computed using (5) and illustrated in the Fig. 3 (*left* graph). Observe that the blue curve coincides with the red one at the beginning and gradually moves towards the green one, implying that frame confidences determine the contributions of frames, i.e., the frame assigned with a large (small) confidence has a large (small) influence on the final trajectory. It is worth pointing out that the confidence-weighted scheme provides a straightforward way to incorporate additional human experience (if available) about the importance of task frames, whereas the original formulation of TP-GMM does not address this functionality.

B. Task Frame Optimization

Here we show the experiments corresponding to the optimization of task parameters (*Algorithm* 1), where the same set of demonstrations of the reaching skill introduced in Section VI-A are used. The optimization is carried out under three different scenarios: (*i*) reaching a pole (Section VI-B.1), (*ii*) reaching a point (Section VI-B.2), and (*iii*) obstacle avoidance (Section VI-B.3). In task (*i*), we compare our approach with TP-GMM [4] to show the importance of optimizing task frames. In task (*ii*), we compare our work with optimization of GMM [5] to show the efficiency of our method. In task (*iii*) we show that obstacle avoidance can be achieved by optimizing task parameters.

1) Reaching a Pole : We consider the reaching of a static pole that is located at $[0.3\ 0.1]$ with the height ranging from -0.2 to 0.4, and the robot only needs to reach it regardless of the exact location along the vertical axis. Here, we introduce an additional constraint in the joint space, which minimizes the weighted joint displacement, i.e.,

$$f_q = \sum_{t=1}^{N-1} ||\mathbf{W}(\mathbf{q}_{t+1} - \mathbf{q}_t)|| \qquad (12)$$

where $\mathbf{W}$ denotes a weight matrix and $||\cdot||$ represents 2-norm. In order to illustrate the importance of optimizing task parameters, we take TP-GMM as a comparison. Since the robot starts from the same position, we define frame $\{1\}$ with parameters $\mathbf{A}^{(1)} = \mathbf{I}_{4\times4}$, $\mathbf{b}^{(1)} = [0\ 0.005\ 0.156\ \ -0.050]^T$. Note that the reaching point can be located at any position along the pole vertical axis, so we define three groups of task parameters for frame $\{2\}$, as shown in Table I, which are compared with the optimized task parameters later. In contrast to these manually defined parameters, we employ our approach (introduced in Section IV) to search optimal task parameters (i.e., the rotation operation and the vertical component in the translational operation) for frame $\{2\}$, where 500 trials are carried out and the policy parameters in (8) are updated every 10 trials. The optimal task parameters, found by our approach, are presented in Table I. The Cartesian trajectories that are generated by using task parameters from Table I are depicted in Fig. 4. The cost values of using task parameters from group 1, group 2 and group 3 are 5.21, 5.87 and 5.70, respectively. The optimal task parameters have the cost value 3.26. As it can be seen, it is difficult to manually define appropriate task parameters in TP-GMM. Instead, our method provides an effective way to set task parameters while addressing additional requirements.

2) Reaching a Point : In this task, we consider the same joint constraint (12). Now, let us first consider the reaching of an object at $[0.3\ 0.3\ 0.2]^T$ in frame $\{O\}$. We define two task frames described by $\mathbf{A}^{(1)} = \mathbf{A}^{(2)} = \mathbf{I}_{4\times4}$, $\mathbf{b}^{(1)} = [0\ 0.005\ 0.156\ \ -0.050]^T$, $\mathbf{b}^{(2)} = [0\ 0.3\ 0.3\ 0.2]^T$. Note that both frames have their origins at the start and target points respectively, thus only the rotation operations are implemented. We use our approach to learn rotation angles for both frames simultaneously, where the policy parameters are updated every 10 roll-outs. For comparison purposes, we also evaluate the optimization of GMM components using PI^2. Here, similar to [5], we optimize Gaussian means while the higher dimensional covariance matrices are kept fixed. Besides this current target, we evaluate both optimizations again using a different target located at $[-0.3\ \ 0.2\ \ 0.2]^T$ in $\{O\}$. We have 5 runs for each method and for each target. Meanwhile, we calculate the average cost every 20 roll-outs in each run. Finally, the means and standard deviations of average costs are computed, as shown by the

TABLE I

TASK PARAMETERS OF FRAME $\{2\}$

	Group 1	Group 2	Group 3	Optimized Parameters
$\mathbf{A}^{(2)}$	$\mathbf{I}_{4\times4}$	$\begin{bmatrix}1&0&0&0\\0&0&-1&0\\0&1&0&0\\0&0&0&1\end{bmatrix}$	$\begin{bmatrix}1&0&0&0\\0&0&1&0\\0&-1&0&0\\0&0&0&1\end{bmatrix}$	$\begin{bmatrix}1&0&0&0\\0&0.5477&-0.6153&-0.5670\\0&0.7500&0.6615&0.0067\\0&0.3709&-0.4288&0.8237\end{bmatrix}$
$\mathbf{b}^{(2)}$	$[0\ 0.3\ 0.1\ 0.3]^T$	$[0\ 0.3\ 0.1\ 0.05]^T$	$[0\ 0.3\ 0.1\ 0]^T$	$[0\ 0.3\ 0.1\ 0.14]^T$

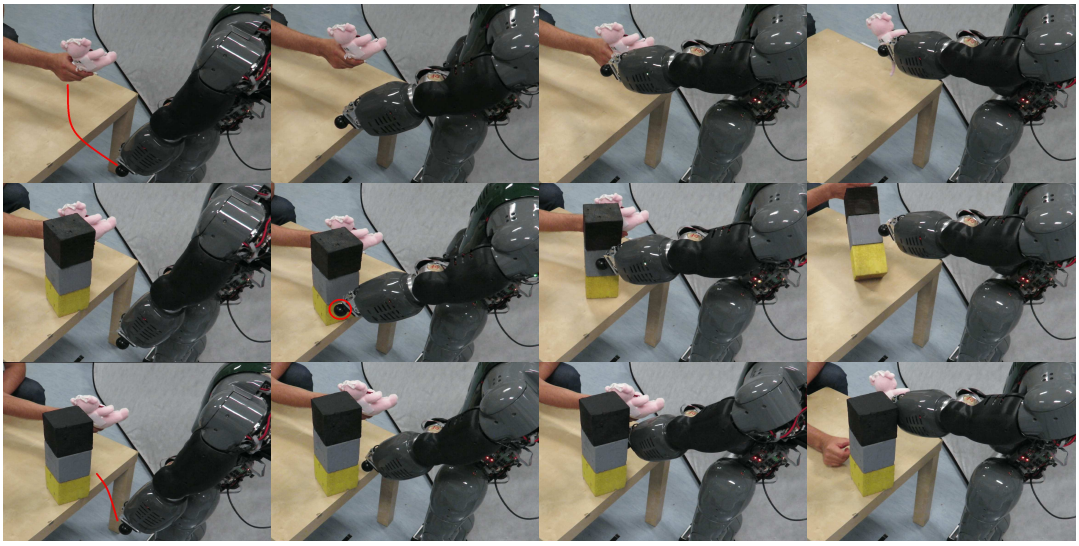

Fig. 6. *Top row*: reaching task learned by optimizing task parameters with respect to the cost function (12). *Middle row*: reaching task with obstacle collision, where only the constraint (12) is used. *Bottom row*: reaching task learned by optimizing task parameters with respect to the cost function (13).

error-bar curves in Fig. 5. Our approach converges faster than GMM components optimization. This result coincides with our intuitions since the frame-based optimization has fewer parameters compared with the GMM optimization. In addition to these evaluations, we test the reaching task on the real COMAN robot, as shown in Fig. 6 (*top* row), showing that the proposed algorithm generates a trajectory that allows COMAN to perform successfully.

3) Obstacle Avoidance: We here consider the case in which an obstacle occupies a portion of the learned robot movement path (shown in Fig. 6). In order to formulate the cost function easily, we simplify the obstacle as a bounded rectangle S and the robot end-effector as a point. Subsequently, we estimate the intersection point $\tilde{\mathbf{p}}$ of the end-effector trajectory and S, and determine if the intersection point lies inside or outside S. Furthermore, let us denote the distance between $\tilde{\mathbf{p}}$ and each edge of S as d_i with $i \in \{1,2,3,4\}$. Then, the cost function can be defined as

$$C = \begin{cases} f_q + k_1 e^{(k_2 d)}, & \tilde{\mathbf{p}} \in S \\ f_q + k_3 e^{(-k_4 d)}, & \tilde{\mathbf{p}} \notin S \end{cases}, \tag{13}$$

where $d = \min\{d_1, d_2, d_3, d_4\}$ and $k_i > 0$. Here, the joint constraint is used to avoid large trajectory deviation from the original desired trajectory. As a comparison, we test the reaching task using only the cost function (12). The evaluations on the real robot are illustrated in Fig. 6 (*middle* and *bottom* rows), showing that the robot is capable of both avoiding the obstacle and reaching the target object by optimizing task parameters with respect to (13).

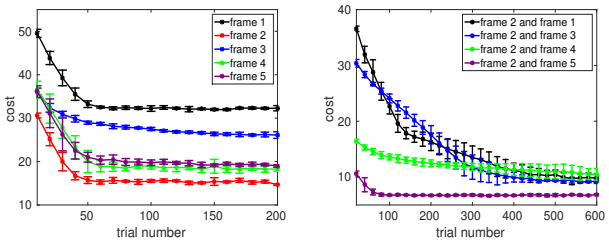

Fig. 7. These figures depict cost values in the frame selection. The *left* plot shows cost values through optimizing a single task frame while the *right* plot shows cost values of optimizing the best frame $\{2\}$ and each of the rest frames. Error-bars represents means and standard deviations of cost values.

C. Automatic Frame Selection

We have evaluated a fixed set of task frames in the experiments previously reported, now we consider a transportation task to show the application of the proposed frame selection scheme given a large set of candidate frames. We collected 8 demonstrations of the task through kinesthetic teaching, which guided the robot to reach and pick up an object, and subsequently release it at the goal position, lasting about $10s$ each. We defined 5 initial candidate frames associated with the end-effector positions at time steps $2s$, $4s$, $7s$, $9.5s$

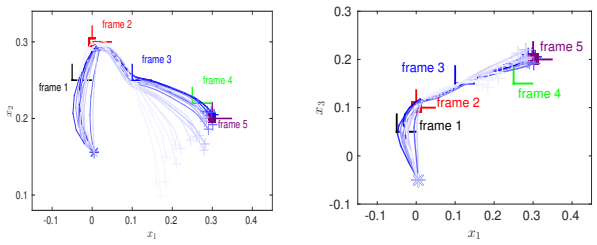

Fig. 8. These figures show the trajectory evolutions (color: from light to dark) through optimizing task parameters of frame $\{2\}$ and frame $\{3\}$ simultaneously. The start and end point of each trajectory are depicted by the blue '$*$' and '+', respectively. The red and purple solid boxes denote the desired via-point and end-point.

and $10s$, respectively. Then, we projected the demonstrations into these candidate frames, and subsequently trained local GMMs and generated local trajectories.

Considering a new task instance, which requires the robot to pick up the object located at $\mathbf{p}_s = [0\ 0.3\ 0.1]^T$ (in frame $\{O\}$) when $t = 4s$, and subsequently release it at $\mathbf{p}_e = [0.3\ 0.2\ 0.2]^T$ (in frame $\{O\}$) when $t = 10s$. Meanwhile, we expect to reduce the joint displacement. Through combining the task and joint constrains, the cost function is defined as

$$C = f_q + k_{p1}||\mathbf{p}_{t=4} - \mathbf{p}_s|| + k_{p2}||\mathbf{p}_{t=10} - \mathbf{p}_e||, \qquad (14)$$

where k_{p1} and k_{p2} are positive scalars. For this new task, we adapt 5 candidate frames using new task parameters $\mathbf{A}^{(j)} = \mathbf{I}_{4\times4}$, $j = \{1, 2, \ldots, 5\}$, $\mathbf{b}^{(1)} = [0\ \ -0.05\ \ 0.25\ \ 0.05]^T$, $\mathbf{b}^{(2)} = [0\ \ 0.0\ \ 0.3\ \ 0.1]^T$, $\mathbf{b}^{(3)} = [0\ \ 0.10\ \ 0.25\ \ 0.15]^T$, $\mathbf{b}^{(4)} = [0\ \ 0.25\ \ 0.22\ \ 0.15]^T$ and $\mathbf{b}^{(5)} = [0\ \ 0.3\ \ 0.2\ \ 0.2]^T$. Since these candidate frames only differ in their origins, it is not needed to apply translational operations, and hence we focus on rotation operations. We first evaluate each frame separately as shown in Fig. 7 (*left* plot). Since the frame $\{2\}$ has the most significant influence on the pick-and-place task (i.e., the smallest cost values and the fastest convergence speed), it is viewed as the most important frame. Furthermore, we evaluate the combination of frame $\{2\}$ and the rest of frames. An illustration of trajectory evolutions in one run through combined optimization of frame $\{2\}$ and $\{3\}$ is reported in Fig. 8. The evaluations of two frames are shown in Fig. 7 (*right* graph), showing that the combined performance of frame $\{2\}$ and frame $\{5\}$ attains the lowest cost values. In summary, the frame forward search provides an optimal solution to define a frame set achieving lowest cost values. Finally, we emphasize that humans usually have better understanding of task goals than task frames, and thus the strategy of frame selection offers an alternative solution to discover the most task-relevant frames automatically.

VII. Conclusions

In this paper we presented a generalized task-parameterized learning framework, which is initially learned from human demonstrations. The generalization first considers the confidence-weighted scheme, which allows for the incorporation of human prior knowledge on the task frames into a variant of TP-GMM. Subsequently, a novel learning perspective is proposed, which directly optimizes task parameters instead of GMM components, rendering a lower dimensional optimization problem. Moreover, an iterative feature selection scheme is proposed, which has shown effective to select important task frames and remove frames that are either redundant or irrelevant for the task. In our evaluations, we learn task parameters of different frames without considering their correlations. However, in many tasks (e.g., the robot bimanual task) the task frames are often relevant to each other, and thus the correlations between frames could be exploited, which might help to accelerate the learning process. In addition, since various movement primitives such as non-parametric [17] and parametric [18] formulations have been developed, a comprehensive comparison needs further exploitation.

References

[1] B. D. Argall, S. Chernova and M. Veloso, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469-483, 2009.

[2] C. G. Atkeson, A. W. Moore, S. Schaal, "Locally weighted learning," *Artificial Intelligence Review*, vol. 11, pp. 11-73, 1997.

[3] C. E. Rasmussen, C. K. I. Williams, Gaussian Processes for Machine Learning. The MIT Press, 2006.

[4] S. Calinon, D. Bruno and D. G. Caldwell, "A task-parameterized probabilistic model with minimal intervention control," in *Proc. IEEE International Conference on Robotics and Automation*, 2014, pp. 3339-3344.

[5] F. Guenter, M. Hersch, S. Calinon, "Reinforcement learning for imitating constrained reaching movements," *Advanced Robotics*, vol. 21, no. 13, pp. 1521-1544, 2007.

[6] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural Computation*, vol. 25, no. 2, pp. 328-373, 2013.

[7] L. Rozo, S. Calinon, D. G. Caldwell, "Learning physical collaborative robot behaviors from human demonstrations," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 513-527, 2016.

[8] J. Silvério, L. Rozo, S. Calinon, "Learning bimanual end-effector poses from demonstrations using task-parameterized dynamical systems," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 464-470.

[9] M. Muhlig, M. Gienger, S. Hellbach, J. J. Steil and C. Goerick, "Task-level imitation learning using variance-based movement optimization," in *Proc. IEEE International Conference on Robotics and Automation*, 2009, pp. 1177-1184.

[10] D. A. Cohn, Z. Ghahramani, M. I. Jordan, "Active Learning with Statistical Models," *Journal of Artificial Intelligence Research*, vol. 4, no. 1, pp. 129-145, 1996.

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

[12] T. Alizadeh, *Statistical learning of task modulated human movements through demonstration*, University of Genova, PhD thesis, 2014.

[13] J. Buchli, F. Stulp, E. Theodorou, "Learning variable impedance control," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 820-833, 2011.

[14] E. Theodorou, J. Buchli, S. Schaal, "A generalized path integral control approach to reinforcement learning," *Journal of Machine Learning Research*, vol. 11, pp. 3137-3181, 2010.

[15] F. Stulp and O. Sigaud, "Robot skill learning: from reinforcement learning to evolution strategies," *Journal of Behavioral Robotics*, vol. 4, no. 1, pp. 49-61, 2013.

[16] N. G. Tsagarakis, S. Morfey and G. M. Cerda, "Compliant humanoid COMAN: optimal joint stiffness tuning for modal frequency control," in *Proc. IEEE International Conference on Robotics and Automation*, 2013, pp. 673-678.

[17] Y. Huang, L. Rozo, J. Silvério and D. G. Caldwell, "Kernelized movement primitives," *arXiv*:1708.08638, 2017.

[18] A. Paraschos, C. Daniel, J. Peters and Neumann G, "Probabilistic movement primitives," in *Proc. Advances in Neural Information Processing Systems*, 2013, pp. 2616-2624.