Learning Task-Parameterized Skills from Few Demonstrations

Jihong Zhu, Michael Gienger, Jens Kober

2022

Preprint:

This is an accepted article published in ICRA / RA-L. The final authenticated version is available online at: https://doi.org/10.1109/LRA.2022.3150013

Learning Task-Parameterized Skills from Few Demonstrations

Jihong Zhu, *Member, IEEE*, Michael Gienger, and Jens Kober, *Senior Member, IEEE*

***Abstract*—Moving away from repetitive tasks, robots nowadays demand versatile skills that adapt to different situations. Task-parameterized approaches improve the generalization of motion policy by encoding relevant contextual information in the task parameters, hence enabling flexible task executions. However, training such a policy often requires collecting multiple demonstrations in different situations. To create these situations, objects or sometimes even humans need to move around, which renders method the less applicable to real-world problems. Therefore, training with fewer demonstrations/situations is desirable. In the paper, we utilize task parameters to generate new demonstration data that augments the original training dataset for policy improvements, thus allows learning task-parameterized skills with few demonstrations.**

***Index Terms*—**

I. Introduction

In contrast to industrial robots that operate in cages and perform repetitive tasks, a next generation of robots is expected to have higher autonomy, the ability to operate in unstructured environments and to be adaptive in task executions. Learning from demonstration (LfD) is a promising step in this direction, enabling robots to acquire versatile motor skills without explicitly programming the motion, thus facilitating robot skill learning.

In LfD, robot motion policies are generated from an underlying model that is trained from demonstration data. How to use the data efficiently and produce policies that generalize well to new situations is at the core of robot LfD research [1]. One prominent example, Task-Parameterized Gaussian Mixture Models (TP-GMM) improves generalization by encoding the task-relevant states into the task parameters and use them for generating motions in a new situation [2]. In TP-GMM, the task parameters are reference frames that describe the spatial configurations of the situation. Perspectives from different reference frames are leveraged to produce a policy that adapts to the current situation.

Multiple demonstrations in different situations need to be collected for obtaining the TP-GMM. Hence, the collected observation data needs to comprise many different spatial configurations of the task to provide enough statistics for a meaningful model. This is often impracticable in practice, e.g., in a factory or household environment. Furthermore, demonstrating the task with changing parameters is more likely to introduce ambiguity in the demonstration. For instance, if there is an object that the robot needs to avoid during task execution, in TP-GMM, a reference frame will be assigned to the object. During demonstrations, it is not easy to ensure that the demonstrator always goes from the similar direction in the object frame for avoidance, thus bring in ambiguity and consequently compromise the policy [3].

The contribution of this paper is a concept for learning task-parameterzied skills from few demonstrations. Instead of solely imitating the expert, it allows generation of synthetic demonstration data that aggregate the original dataset for improving the TP-GMM. The framework reduces the number of demonstrations needed for training task-parameterized skills, improves the data efficiency, and subsequently trims the possibility of ambiguous demonstrations, thus making the task-parameterized skill learning more appealing in practice.

In the next section, we review related works about LfD with a focus on task-parameterized learning. A brief description of the TP-GMM is presented in Sect. III. In Sect. IV, we describe our algorithm. The algorithm is then discussed and validated with simulation in Sect. V. Sect. VI showcases our algorithm on a robotic dressing assistance task. Finally in Sect. VII, we conclude.

II. Related Works

Methods such as dynamical movement primitives (DMP) [4], probabilistic movement primitives (ProMP) [5] and Gaussian mixture models (GMMs) [6] have been used for encoding movements with LfD. These methods are known for data efficiency and can generate robot motion policy from a small number of demonstrations. Alternatively, instead of relying on expert demonstrations, reinforcement learning (RL) generates data by random exploration and finds an optimal policy by reward optimization. This is usually less data efficient than LfD-based approaches. Nevertheless, combining LfD and RL are reported to further boost data efficiency [7], [8].

LfD benefits from demonstration data for skill learning, nevertheless, it is also limited by the reliance on data [1]. Thus, numerous research has been focused on algorithmic development that increases the generalization in LfD. For instance, [9] improves original DMP by selecting a Hilbert norm that reduces the deformation in the retrieved trajectory. In more recent research, [10] combines the DMP and ProMP and proposes Viapoints Movement Primitives that outperforms ProMP in extrapolation. Task parameterized models are alternative approaches for better generalization of the policy. In these approaches, the contextual information about the task are described by task parameters, and movement is encoded with

J. Zhu is with Cognitive Robotics, 3mE, Delft University of Technology and Honda Research Institute, Europe

M. Gienger is with Honda Research Institute, Europe

J. Kober is with Cognitive Robotics, 3mE, Delft University of Technology

either GMMs or hidden Markov models (HMMs) [2], [11], [12]. By combining the movement model with task parameters, TP-GMM(HMM) can produce a policy that adapts to different situations.

Multiple improvements on the original TP-GMM have been made in previous research. By learning the forcing term in DMP with TP-GMM, [11] presents an approach that resolves the divergence problem usually associated with GMM type models. [12] introduces weighting to TP-GMM and [13] build on the idea and propose to infer weights from co-variances in the normal distribution.

While the focus of above-mentioned improvements is on better policy generalization instead of data efficiency, our paper alternatively addresses the latter which is an equally important problem in LfD.

III. Preliminaries

Below we briefly present the TP-GMM algorithm, and for an in-depth tutorial on the subject, the reader can refer to [2]. We split the TP-GMM into two phases: *Model training* and *Fusion for new situations*, where the former describes how to obtain a TP-GMM from demonstrations, and the latter applies the TP-GMM to new situations.

A. Model Training:

In TP-GMM, the situation states for the task is described using N references frames:

$$\left\{\boldsymbol{A}_n, \boldsymbol{b}_n\right\}_{n=1}^{N} \tag{1}$$

with $\boldsymbol{A}_n \in SO(p)$, $\boldsymbol{b}_n \in \mathbb{R}^p$ represents the orientation and displacement in n^{th} reference frame of p dimensions[1].

The demonstrated motion is denoted as $\boldsymbol{\xi}$ and composed of input and output data:

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\xi}^{\mathcal{I}} \\ \boldsymbol{\xi}^{\mathcal{O}} \end{bmatrix}. \tag{2}$$

For a time-based TP-GMM, the input is one dimensional time, and the outputs are the positions, while a trajectory-based TP-GMM has positions as inputs and velocities (and maybe also accelerations) as outputs.

Given M demonstrated trajectories $\{\boldsymbol{\xi}_m\}_{m=1}^{M}$, for the m^{th} demonstration, we assign the corresponding N reference frames: $\{\boldsymbol{A}_{m,\,n}, \boldsymbol{b}_{m,\,n}\}_{n=1}^{N}$ to the demonstration. Using these reference frames, we can transform each demonstration into N frames. Once done, we have a dataset consisting demonstrated trajectories seen from N frames.

A TP-GMM with K components can be trained from this dataset and is defined by:

$$\left\{\pi_k, \{\boldsymbol{\mu}_k^n,\ \boldsymbol{\Sigma}_k^n\}_{n=1}^{N}\right\}_{k=1}^{K}, \tag{3}$$

where π_k is the k^{th} mixing coefficient, and $\boldsymbol{\mu}_k^n$, $\boldsymbol{\Sigma}_k^n$ are respectively the center and covariance matrix of the k^{th} Gaussian component in frame n.

[1]In robotic, depending on the task, $p = 2$ or 3.

B. Fusion for New Situations

For a new situation defined by a set of N references frames:

$$\{\hat{\boldsymbol{A}}_n, \hat{\boldsymbol{b}}_n\}_{n=1}^{N}, \tag{4}$$

a GMM that produces the motion in the new situation can be derived using (3) and (4) in two steps.

Step 1, for n^{th} new reference frame, we transform the Gaussian distributions in n^{th} frame in (3) into the new frame using (4):

$$\begin{aligned} \hat{\boldsymbol{\mu}}_k^n &= \hat{\boldsymbol{A}}_n \boldsymbol{\mu}_k^n + \hat{\boldsymbol{b}}_n \\ \hat{\boldsymbol{\Sigma}}_k^n &= \hat{\boldsymbol{A}}_n \boldsymbol{\Sigma}_k^n \hat{\boldsymbol{A}}_n^T \end{aligned} \tag{5}$$

We apply (5) to all K components in (3).

Step 2, for every component, we fuse the transformed Gaussian distributions in N frames (which we obtained from (5) in step 1) by:

$$\hat{\boldsymbol{\Sigma}}_k^{-1} = \sum_{n=1}^{N} \hat{\boldsymbol{\Sigma}}_k^{n^{-1}},\ \ \hat{\boldsymbol{\mu}}_k = \hat{\boldsymbol{\Sigma}}_k \sum_{n=1}^{N} \hat{\boldsymbol{\Sigma}}_k^{n^{-1}} \hat{\boldsymbol{\mu}}_k^n, \tag{6}$$

The new GMM that adapt to the new frames is then $\{\pi_k, \hat{\boldsymbol{\mu}}_k, \hat{\boldsymbol{\Sigma}}_k\}_{k=1}^{K}$. Likewise in (2), the resulting GMM can be decomposed as:

$$\hat{\boldsymbol{\mu}}_k = \begin{bmatrix} \hat{\boldsymbol{\mu}}_k^{\mathcal{I}} \\ \hat{\boldsymbol{\mu}}_k^{\mathcal{O}} \end{bmatrix},\ \ \hat{\boldsymbol{\Sigma}}_k = \begin{bmatrix} \hat{\boldsymbol{\Sigma}}_k^{\mathcal{I}} & \hat{\boldsymbol{\Sigma}}_k^{\mathcal{I}\mathcal{O}} \\ \hat{\boldsymbol{\Sigma}}_k^{\mathcal{O}\mathcal{I}} & \hat{\boldsymbol{\Sigma}}_k^{\mathcal{O}}, \end{bmatrix} \tag{7}$$

Given input data $\boldsymbol{y}^{\mathcal{I}}$ (current time instance for time-based TP-GMM or positions for trajectory-based TP-GMM) The motion generation in the new situation is solved as a conditional distribution problem using Gaussian Mixture Regression (GMR) [2].

IV. Methods

We present the proposed method comprehensively in this section. We start by giving an intuitive description in Sect. IV-A then an in-depth explanation on each step of the algorithm in Sect. IV-B.

A. Design Considerations

Rather than explicitly minimizing the difference between learned and demonstrated policy, TP-GMM relies on a good representation of data distributions in each reference frame in order to obtain a good policy in different situations. The distribution is learned with GMMs by stacking the demonstration data in each frame together. Individual GMMs for each frame is then extracted from the learned GMMs and fused for assigning the relevant frame to various parts of the movement.

In the model training step presented in Sect. III-A, the learned TP-GMM in the form of (3) maximizes the joint likelihood of data distributions in different reference frames. Since we start with only few demonstrations, the data will be sparse, and the learned model will not be able to capture the distributions well in each frame. Subsequently, the model would fail to obtain a decent fusion for policy generation.

By explicitly taking the production error as the selection criteria, our algorithm generates and select synthetic data that augment the original training data for policy improvement, thus allows learning task-parameterzied skills from few demonstrations.

B. Learning Task Parameterized Skills from Few Demonstrations

We design Alg. 1 under the premise for learning task-parameterized skill with few demonstrations. We elaborate on the steps marked in bold in this section.

Algorithm 1 Learning Task-Parameterized Skills from Few Demonstrations

Inputs: Initial μ numbers of demonstrations, $\mathcal{D}_{\text{init}}$
Maximum number of demonstrations, M
Maximum number of iteration, L
Outputs: Final TP-GMM $\mathcal{P}_{\text{final}}$

Train a TP-GMM $\mathcal{P}$ based on $\mathcal{D}_{\text{init}}$
Cost Computation: $\mathcal{J}(\mathcal{P}, \mathcal{D}_{\text{init}})$
iter $= 0$, $n_{\text{d}} = \mu$, $\mathcal{D} = \mathcal{D}_{\text{init}}$
while $n_{\text{d}} \leq M$ and iter $\leq L$ **do**
  Generate Synthetic Data $\mathcal{D}_n$
  Aggregrate datasets: $\mathcal{D}' \leftarrow \mathcal{D} + \mathcal{D}_n$
  Retrain the TP-GMM $\mathcal{P}'$ from $\mathcal{D}'$
  Compute the cost with new TP-GMM $\mathcal{J}'(\mathcal{P}', \mathcal{D}_{\text{init}})$
  if $\mathcal{J}' < \mathcal{J}$ **then**
    $\mathcal{D} = \mathcal{D}'$
    $\mathcal{P} = \mathcal{P}'$
    $n_{\text{d}} = n_{\text{d}} + 1$
  end if
  iter $=$ iter $+ 1$
end while
$\mathcal{P}_{\text{final}} = \mathcal{P}$
return $\mathcal{P}_{\text{final}}$

Inputs: In this step, we mainly discuss demonstrations collection. We collect μ numbers of demonstrations (where $\mu \geq 2$) $\mathcal{D}_{\text{init}}$ for the initial training of the TP-GMM. In order to avoid that the final policy over-fits to some local configurations and improve generalization, it is better if the demonstrations are collected in distinctive situations (see Sect. VI for some discussions.). Since in TP-GMM, the situations are described with reference frames, the distinctive measure is the distances between the corresponding reference frames, and the angles that represents difference in orientations. Training a TP-GMM from initial demonstrations is a standard procedure described in Sect. III.

Cost Computation: The cost is defined as the reproduction error between the TP-GMM produced policy and the expert demonstration. Therefore it takes the initial demonstration $\mathcal{D}_{init}$ and the TP-GMM $\mathcal{P}$ as inputs.

For time-based TP-GMM, the cost is defined as root mean square error. For μ number of initial demonstrations, we represent the cost as:

$$\mathcal{J}_{\text{RMS}} = \frac{1}{\mu}\sum_{i=1}^{\mu}\sqrt{||\boldsymbol{y}_i - \boldsymbol{\xi}_i||}, \tag{8}$$

where $\boldsymbol{y}_i$ is the reproduction of initial demonstrations from the TP-GMM and $\boldsymbol{\xi}_i$ is the initial expert demonstration, both in the i^{th} instance.

For trajectory-based TP-GMM, we define the cost of the TP-GMM as the normalized distance computed by dynamic time warping (DTW) between the reproduction and the expert demonstrations.

Give the reproduction $\boldsymbol{y} \in \mathbb{R}^N$ and corresponding expert demonstration $\boldsymbol{\xi}_d \in \mathbb{R}^M$ (note for trajectory-based TP-GMM, N does not necessarily equals M), DTW tries to find a warping function $\phi(K)$, where $K = 1, 2, \cdots, T$:

$$\phi(K) = \big(\phi_{\boldsymbol{y}}(K), \phi_{\boldsymbol{\xi}}(K)\big),$$

where $\phi_{\boldsymbol{y}}(K) \in \boldsymbol{y}$, and $\phi_{\boldsymbol{\xi}}(K) \in \boldsymbol{\xi}$. The warping function maps the data in $\boldsymbol{y}$ to the data in $\boldsymbol{\xi}$ with minimum dissimilarity d. The dissimilarity d between two data points is defined as their Euclidean distance. The cost function for DTW is defined as the average accumulated dissimilarity between warped $\boldsymbol{y}$ and $\boldsymbol{\xi}$:

$$d_\phi(\boldsymbol{y}, \boldsymbol{\xi}) = \sum_{K=1}^{T} d(\phi_{\boldsymbol{y}}(K), \phi_{\boldsymbol{\xi}}(K))\zeta_\phi(K)/Z, \tag{9}$$

where $\zeta_\phi(K)$ is a weighting coefficient and Z is the normalization constant (we refer readers to [14] for a detailed definition). The warping function is computed by minimizing the cost defined in (9):

$$D(\boldsymbol{y}, \boldsymbol{\xi}) = \min_{\phi} d_\phi(\boldsymbol{y}, \boldsymbol{\xi}) \tag{10}$$

For μ number of initial demonstrations (each with a distinctive situation), we represent the cost for trajectory-based TP-GMM as:

$$\mathcal{J}_{DTW} = \frac{1}{\mu}\sum_{i=1}^{\mu} D(\boldsymbol{y}(i),\ \boldsymbol{\xi}_d(i)) \tag{11}$$

We select the cost depending on the type of TP-GMM and use it for deciding whether we should update the dataset and TP-GMM or not.

Generate Synthetic Data: In this step, we generate new motion data to augment original demonstration for policy improvement. We present 3 different data generation methods for Alg. 1 and introduce each subsequently:

- *Method 1:* Injecting white noise to the original expert demonstrations (Fig. 1b),
- *Method 2:* TP-GMM generated data random new situations (Fig. 1c),
- *Method 3:* TP-GMM generated data under random new situations adding white noise (Fig. 1d),

For *Method 1*, we consider injecting white noise into the expert demonstration for generating the synthetic data. Noise injection on the training data is known method for improving learning outcomes [15]. It has been applied on Deep Neural Networks (DNNs) for achieving better generalization capability [16] and robustness against adversarial attacks [17]. Policy improvement can also be made from single demonstration that augmented with noise [18]. In robotics LfD, DART was proposed for robust imitation learning [19]. Recently, the authors of [20] present a method that adds different level of noise into the demonstration data for improving the reward learning in a inverse reinforcement learning setting.

Method 2 generates random reference frames that satisfies all task constraints such as kinematic limits to create a new

situation. Then applies the TP-GMM to the new situation for generation of new demonstration data.

In TP-GMM, reference frames are attached to movable entities that are relevant for the task. Depending on the task space and the property of the entity, the frame orientation and translation are bounded for the task. For instance, orientation of the frames can be expressed in Euler angle with maximum and minimum values:

$$\alpha \in [\alpha_{\min}, \alpha_{\max}],\ \beta \in [\beta_{\min}, \beta_{\max}],\ \gamma \in [\gamma_{\min}, \gamma_{\max}], \quad (12)$$

whereas translation limitation are expressed with:

$$\boldsymbol{b} \in [\boldsymbol{b}_{\min}, \boldsymbol{b}_{\max}]. \quad (13)$$

We perform uniform random sampling between the limits in (12) and (13) to generate reference frames that satisfy kinematic limits of the task. Once we have the new situation, a new motion data $\mathcal{D}_n$ can be obtained from the TP-GMM.

We would like to mention that in the *Method 2*, the new training data is generated as a result of fusion of different Gaussian distributions with new task parameters. It can not be generated by the individual GMM in each frame of the old TP-GMM. Adding the data to the training dataset helps to better understand the distributions in each frame, rather than reinforcing the original policy.

Method 3 is the combination of *Method 1* and *2*. It injects white noise into the data generated from *Method 2* and use it for augmenting the original training data.

We then augment the training dataset with the synthetic data, and retrain a TP-GMM based on the new dataset. Afterwards, the cost $\mathcal{J}$ of the new TP-GMM is compared with the old one. If the cost is reduced, we update the dataset and the TP-GMM. If not, we keep the old dataset and TP-GMM and go back to synthetic data generation step. Note that the cost is only calculated based on the expert demonstrations, and does not include the algorithm generated demonstrations.

Outputs: The algorithm has two termination conditions. The maximum number of the demonstrations in the training dataset $\mathcal{D}$ denoted by M ($M > \mu$) and the number of maximum iterations L. If the algorithm reaches the maximum iteration but fails to add any new demonstrations, one could either sets a larger L, or provides additional demonstrations.

The outputs is the final TP-GMM that has the lowest cost after the dataset augmentation.

V. Simulation and Analysis

In this section, we compare data generation methods proposed in Sect. IV for Alg. 1 in terms of cost reduction and policy generalization on a simulated task. The task considers a 2D movement from the start (the grey U-shape box) to the goal (the yellow U-shape box) as shown in Fig. 1a. The shape imposes constraints for the motion. Three initial demonstrations are collected with different positions and orientations of the target. Fig. 1b, 1c and 1d present the augmented training dataset using *Method 1*, *2* and *3* respectively.

We run the Alg. 1 with the same initial demonstrations (shown in Fig. 1a). The TP-GMM is trained with 8 Gaussians and the termination condition for the algorithm is set to be $M \leq 8$ (maximum number of demonstrations) and $L \leq 100$ (maximum iterations). For the noise injection, the signal to noise ratio (SNR) is set to 30 decibels. We run the algorithm with each data generation methods 20 times and compare their average and maximum cost reduction in Fig 2. We save the model with lowest cost in all three methods and compare policy reproduction and generalization of the models. From Fig. 2 we observe that *method 1* is able to reduce the cost the most. This is reasonable as the augmented data is the original expert demonstration with white noise. *Method 2* and *Method 3* provides compatible cost reduction. As the cost is reduced, they all show better performance in terms of reproduction (see Fig. 3).

Reduction of the reproduction error is one indicator of the performance. More importantly, generalization is the core capability of TP-GMM thus we further investigate the model's generalization capability to new situations. Figure 4 illustrate the generalization capability of the TP-GMM with 4 new situations. Although providing the highest cost reduction, *Method 1* turns to over-fit the demonstration data and magnifies some unwanted movements in the produced motion (cases in the second columns of Fig. 4). While *Method 2* and *Method 3* behave better than the original TP-GMM in all four cases and generates smoother motion than *Method 1*. They produce similar motion and have comparable performance in terms of generalization to new situations.

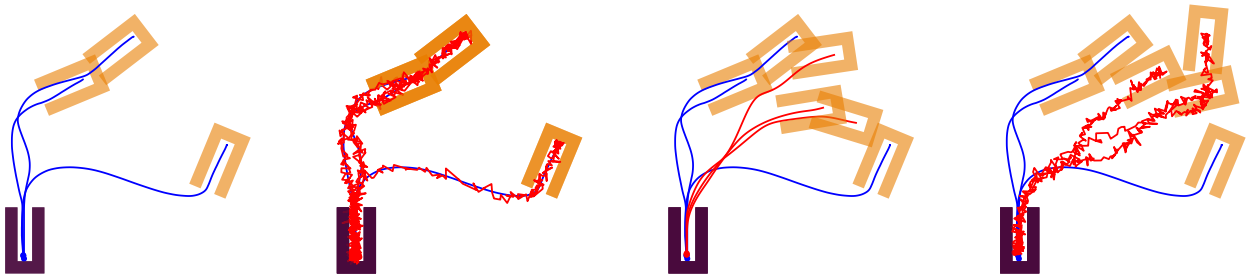
Another way of cost reduction without generating new data is to increase the number of Gaussian K in (3) used in the TP-GMM training phase, however, when initial expert demonstrations are few and sparse, the TP-GMM is likely to over-fit to the expert demonstration, and may magnify unwanted movements in the demonstration likewise in *Method 1*.

In a nutshell, by explicitly using reproduction error as the selection criteria, all three data generation methods are able to generate synthetic motions for data augmentation to improves the final TP-GMM. While TP-GMM produced from *Method 1* tends to over-fit to original demonstration and may magnify unwanted movements, the TP-GMM produced from *Method 2* and *3* are improved over the original TP-GMM in reproduction and also have better generalization capabilities.

VI. Robotic Experiments

We consider the task of dressing a short sleeve shirt onto one arm of a mannequin. We assume the arm posture is static during the dressing and the hand is already inside the armscye. The robot grasped on the shoulder area of the cloth. The dressing starts above the wrist and ends above the shoulder.

The dressing assistance is a primary task that occurs everyday in elderly care, and has been considered in previous assistive robot research. Recently the authors of [21] address the problem from safety perspective and proposed a motion planning strategy that theoretically guarantees safety under the uncertainty in human dynamic models. Zhang et al., uses a hybrid force/position control with simple planning for dressing [22], while [23] use deep reinforcement learning (DRL) to simultaneously train human and robot control policies as



(a) Expert demonstrations

(b) Expert demonstrations augmented with noisy demonstrations. The signal to noise (SNR) level is 30 decibels

(c) Expert demonstrations augmented with generated motion in new situations

(d) Expert demonstrations augmented with generated noisy motion in new situations. The signal to noise (SNR) level is 30 decibels

Fig. 1. Simulated movement tasks with expert demonstrations in blue and synthetic movement data in red: (a). Task representation and original expert demonstration, (b). original expert demonstration and the noisy original demonstrations, (c). original expert demonstration and generated motions in new situations, (d). original expert demonstration and generated noisy motions in new situations.

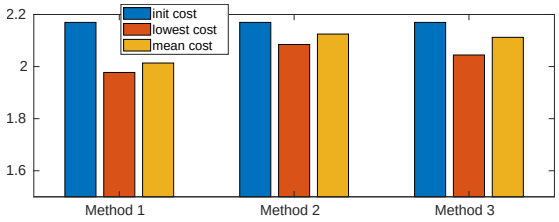
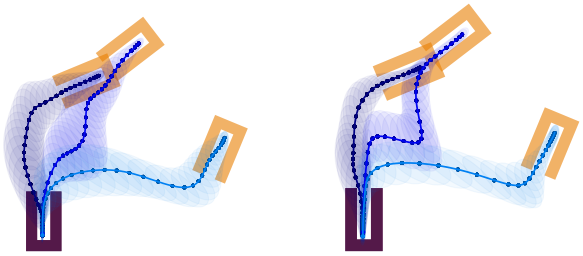
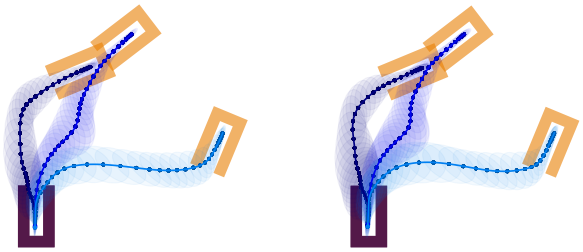


Fig. 2. Cost reduction using different synthetic data generation methods



(a) Reproduction of initial demonstrations from original TP-GMM

(b) Reproduction of initial demonstrations from TP-GMM obtained from *Method 1*



(c) Reproduction of initial demonstrations from TP-GMM obtained from *Method 2*

(d) Reproduction of initial demonstrations from TP-GMM obtained from *Method 3*

Fig. 3. Reproduction of initial demonstration with (a). original TP-GMM, (b). TP-GMM produced with *Method 1*, (c). TP-GMM produced with *Method 2*, (d). TP-GMM produced with *Method 3*. The co-variance of each data point is represented with eclipse centered at the point.

separate neural networks using physics simulations. Although DRL yields satisfactory dressing policies, applying DRL in

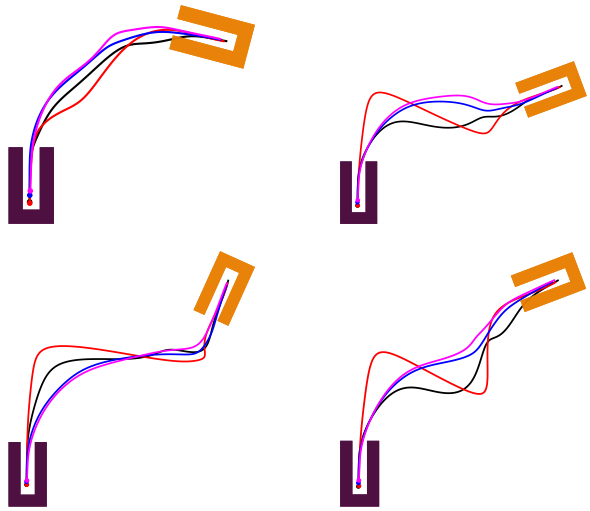


Fig. 4. Generalization to 4 new situations by original TP-GMM (black), TP-GMM produced with *Method 1* (red), TP-GMM produced with *Method 2* (blue), TP-GMM produced with *Method 3* (magenta).

a real world setting is very difficult, especially if the task involves a human. LfD on the other hand, allows programming the robot by non-experts, thus can facilitate dressing skill learning: i.e., the robot can be programmed by healthcare workers.

LfD has been employed to encode dressing policies in the previous research. [24] work with a single static posture and encode the dressing policy by DMP. When consider multiple different static postures and the policy needs to adapt, task-parameterized approaches becomes more relevant. [25] combine sensory information and motor commands as a joint distribution in a hidden semi-Markov model, and then coupled with a task-parameterized model to generalize to different situations. In [26], techniques on incremental learning on the TP-GMM are proposed which allow improvement on the learned policy by new demonstrations. Both of the above-mentioned task-parameterized dressing schemes require several demonstrations (either at the beginning or incrementally) for generalizing the task. In this section, we demonstrate that

using our framework, the robot can learn to dress with only two demonstrations.

During experiments, we record only positions of the end-effector (EE) and the wrist, elbow, shoulder positions during demonstration. The latter is used for calculating the shoulder and wrist reference frames. The robot motion in a new situations is generated by GMR produced from a TP-GMM upon the situation specific task parameters. For simplicity, we fixed the impedance during task execution.

The dressing trajectory needs to adapt to different arm postures. The postures are described with positions of shoulder $\boldsymbol{p}_{\text{sh}}$, elbow $\boldsymbol{p}_{\text{el}}$ and wrist $\boldsymbol{p}_{\text{wr}}$ on a static base frame s which located at the robot base. Two frames are needed to fully describe the posture of an arm. One located at the shoulder, and the other located at the wrist. These two frames constitute the task parameters for the TP-GMM:

$$\{\boldsymbol{A}_{\text{sh}}, \boldsymbol{b}_{\text{sh}}\},\ \{\boldsymbol{A}_{\text{wr}}, \boldsymbol{b}_{\text{wr}}\}, \tag{14}$$

where $\boldsymbol{b}_{\text{sh}} = \boldsymbol{p}_{\text{sh}}$, $\boldsymbol{b}_{\text{wr}} = \boldsymbol{p}_{\text{wr}}$. For two orientations $\boldsymbol{A}_{\text{sh}}$ and $\boldsymbol{A}_{\text{wr}}$, the x axis is defined parallel to vector $\boldsymbol{p}_{\text{sh}}\boldsymbol{p}_{\text{el}}$ and $\boldsymbol{p}_{\text{wr}}\boldsymbol{p}_{\text{el}}$ respectively. Figure 5 shows different postures and their respective references frames at the shoulder and wrist with coordinate depicted with red, green and blue (RGB) arrows.

Since the dressing motion is not explicitly depending on time, we employ the GMM consisting of position and displacement, reference frames can be augmented accordingly to transform both position and displacement components:

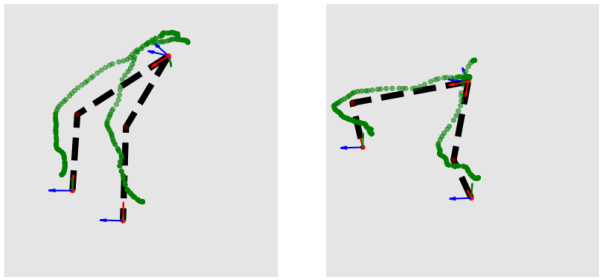
$$\hat{\boldsymbol{p}} = \begin{bmatrix} \boldsymbol{p} \\ \delta\boldsymbol{p} \end{bmatrix} \in \mathbb{R}^6,\ \hat{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A} \end{bmatrix} \in \mathbb{R}^{6\times 6},\ \hat{\boldsymbol{b}} = \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix} \in \mathbb{R}^6 \tag{15}$$

Following the steps described in Sect. III-A, we can obtain a TP-GMM in the form of (3). The dressing motion is generated by integrating the output of the GMR conditioned on the current position.

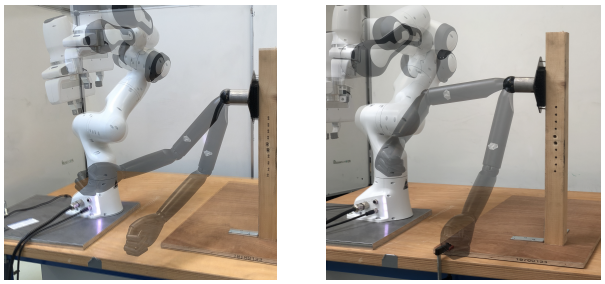
We test our algorithm on two cases. In each case, demonstration dataset contains human demonstrations in two different posture. Figure 5 presents the postures and corresponding demonstrations in both cases.

The maximum number of demonstrations $M = 7$ and maximum iteration $L = 100$ are set for the algorithm. We train the TP-GMM with 4 components. Each initial demonstration dataset are tested on *Method 2* and *Method 3* as the data generation methods. In *Method 3* the SNR is set to be 30 decibels. The cost reduction is indicate in Fig. 6. Two cases have different initial cost as the initial expert demonstrations are different. Both *Method 2* and *Method 3* are able to reduce the cost. Similarly as in the simulation, both methods have compatible performance in terms of cost reduction.

In addition, likewise in Sect. V, we test the generalization capability of the original and the final obtained TP-GMM. These TP-GMM are tested on 5 different postures. We define the success condition for checking the success rate of policy generated from TP-GMM. If a trajectory is able to reach above and around shoulder and does not hit mannequin or stretch the cloth in the dressing process, we consider the dressing successful. The success rate (percentage of success out of total 5 different postures) of the dressing policy generated



(a) case 1 posture and path plot (b) case 2 posture and path plot



(c) Case 1 arm postures for initial demonstrations (d) Case 2 arm postures for initial demonstrations

Fig. 5. Two demonstrations dataset of the dressing task. Each dataset contains demonstrations in two different postures. Arm postures in (a) and (b) characterized by the dashed lines. Demonstrated path are scattered points in green. Local frames are depicted with XYZ (RGB arrows) coordinates at the shoulder and the wrist.

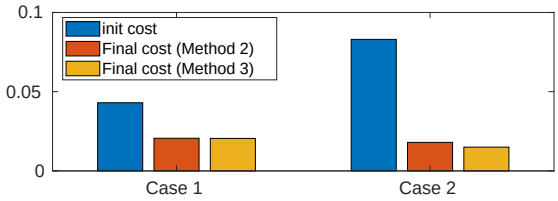


Fig. 6. Cost reduction of case 1 and 2 using proposed algorithm with *Method 2* and *3* as the data generation method. Two cases have different initial cost as the demonstration data are different.

from initial and final (synthetic data augmented) TP-GMM is summarized in Table I. The dressing outcomes for each TP-GMM produced policy is depicts in Fig. 7. Separated by the bold vertical line, the left and right half depicts the end configuration after policy execution from initial and two final TP-GMM (*Method 2* and *Method 3*) for case 1 and 2 respectively.

TABLE I
THE SUCCESS RATE OF INITIAL AND FINAL (SYNTHETIC DATA AUGMENTED) TP-GMM IN TWO CASES AND WITH DIFFERENT SYNTHETIC DATA GENERATION METHODS.

Case 1		Case 2	
init	final	init	final
20%	60% (*Method 2*) 40% (*Method 3*)	20%	80% (*Method 2*) 60% (*Method 3*)

We can observe that case 2 seems to improves more with our method than case 1 in terms of generalization. Take a closer look at the initial demonstrations of both cases, we observe that compared with case 2, in case 1, the initial two postures are more similar to each other and so are the resulting initial

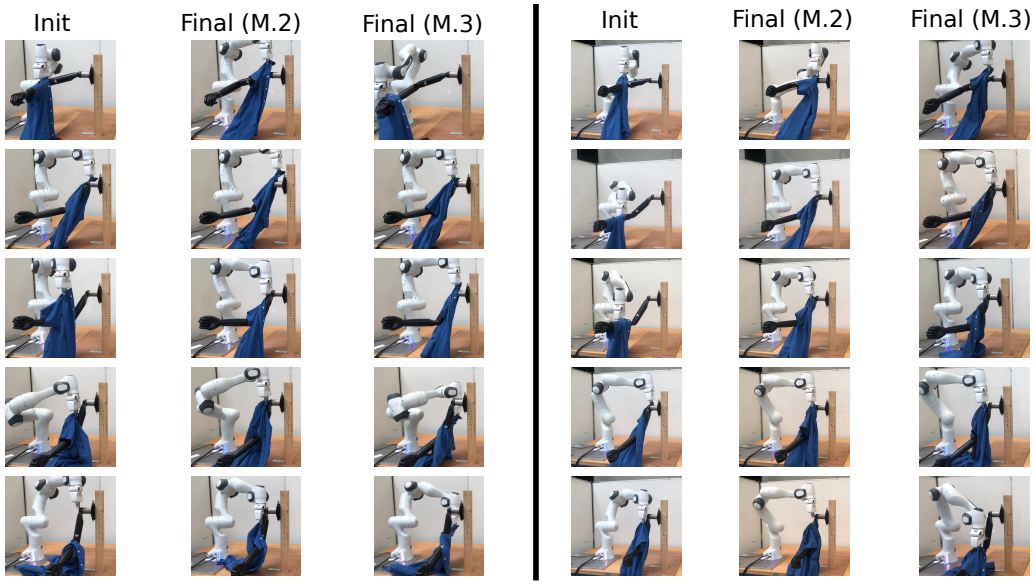
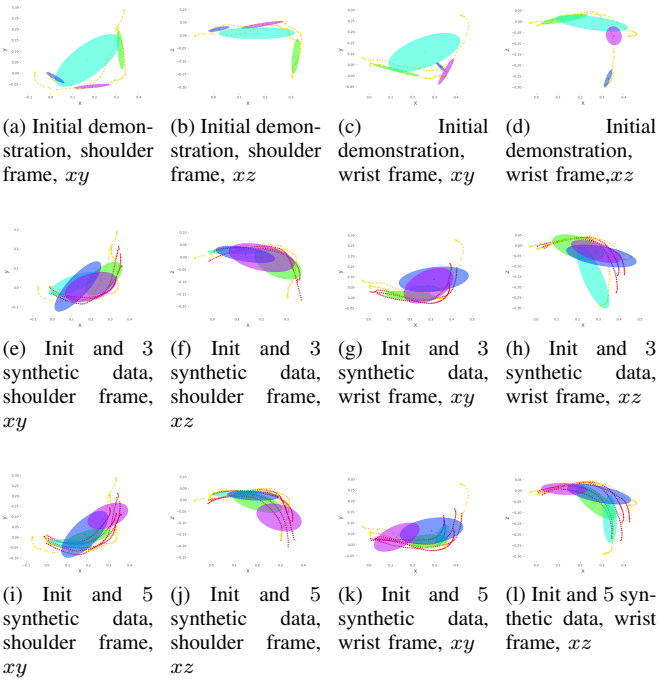


Fig. 7. Dressing outcomes after running the policy from the TP-GMM. Separate by the bold vertical line, the left and right half depicts the end configuration after policy execution from initial and two final TP-GMM (M.2 for *Method 2* and M.3 *Method 3*) for case 1 and 2 respectively.

expert demonstrations. This is also consistent with Fig. 6 that the initial cost of case 1 is less than case 2. The TP-GMM in case 1 is probably over-fit to the data in the region which might explains case 1 performs worse than case 2 in terms of generalization.

We use an example case with the best experiment performance (case 2 with *Method 2* as the data generation method) how added synthetic demonstrations change the distribution of data in each frames in terms of position data, and subsequently the mixture of Gaussian in 2D. We indicate correspondence with the same color in each row. Note that column-wise (which shows the changes of mixture models by adding synthetic data) the same color does not represents correspondence. Rows in Fig. 8 show progressively new demonstration data being added by algorithm and resulting changes of the GMM in each frames. The data colored in yellow are initial expert demonstrations and red are synthetic demonstrations. The first two columns are $3D$ position data in the shoulder frame plotted with xy and xz, third and fourth are the data in the wrist frame. The eclipse represents the co-variance of each multivariate Gaussian.

Table II shows the reduction of the cost defined in (11) for the selected case, after each new synthetic data add to the training dataset. Synthetic data that increased the cost were discarded and not counted. We can also observe that the reduction is larger when the first new demonstration is added, then becomes less significant when the number of demonstrations becomes more.



(a) Initial demonstration, shoulder frame, xy
(b) Initial demonstration, shoulder frame, xz
(c) Initial demonstration, wrist frame, xy
(d) Initial demonstration, wrist frame,xz
(e) Init and 3 synthetic data, shoulder frame, xy
(f) Init and 3 synthetic data, shoulder frame, xz
(g) Init and 3 synthetic data, wrist frame, xy
(h) Init and 3 synthetic data, wrist frame, xz
(i) Init and 5 synthetic data, shoulder frame, xy
(j) Init and 5 synthetic data, shoulder frame, xz
(k) Init and 5 synthetic data, wrist frame, xy
(l) Init and 5 synthetic data, wrist frame, xz

Fig. 8. The synthetic demonstration data being added by algorithm and resulting changes of the GMM in each frames.

VII. Conclusion and Future Works

Task-parameterized approaches often require creating multiple different situations for collecting demonstration thus increase the physical labour during data collection and the risk of having ambiguous demonstrations. We propose an algorithm for learning task-parameterized skills with a reduced number of demonstrations. The algorithm allows generation of new demonstrations that augment the original training dataset for improving the TP-GMM. We validate the algorithm in simulation and on real robot experiments with a dressing assistance task.

We observe that noise injection, creating new situations and the combination of both are possible data augmentation method for improving the TP-GMM. Although noise injection provides highest cost reduction, it has the possibility

TABLE II
COST REDUCTION BY ADDING MORE DEMONSTRATION DATA FROM THE ALGORITHM.

Number of Demonstrations (expert + synthetic)	$2+0$ (init)	$2+1$	$2+2$	$2+3$	$2+4$	$2+5$
Cost	0.0826	0.0407	0.0244	0.0242	0.0203	0.0185

of magnify unwanted movement when few demonstrations are available. The latter two perform similarly in terms of policy improvement while render a better generalization to new situations. In addition, A distinctive initial demonstration may contribute to better generalization performance of the algorithm.

For the dressing task, all TP-GMMs fail to dress all 5 postures. There exists corner cases which probably require additional demonstrations to resolve. That leads to a limitation of the current method: the lack of exploration and feedback. In consequence, the resulting TP-GMM will not able to resolve on the corner cases. Future works may consider tackling the issue by reinforcement learning where the random exploration and feedback (as a reward) can be implemented to compliment current augmentation methods.

Current method considers fix number of Gaussian while as the training data set is augmented, increasing the number of Gaussian will probably result in better approximation of data distribution in each frame and subsequently improve the policy even further.

REFERENCES

[1] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 297–330, 2020.

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

[3] G. Franzese, C. Celemin, and J. Kober, "Learning interactively to resolve ambiguity in reference frame selection," in *Conf. Robot Learning (CoRL)*, 2020.

[4] 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.

[5] A. Paraschos, C. Daniel, J. Peters, G. Neumann, *et al.*, "Probabilistic movement primitives," *Advances in Neural Information Processing Systems*, 2013.

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

[7] J. Kober and J. Peters, "Policy search for motor primitives in robotics," in *Learning Motor Skills*. Springer, 2014, pp. 83–117.

[8] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6292–6299.

[9] A. D. Dragan, K. Muelling, J. A. Bagnell, and S. S. Srinivasa, "Movement primitives via optimization," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2339–2346.

[10] Y. Zhou, J. Gao, and T. Asfour, "Learning via-point movement primitives with inter-and extrapolation capabilities," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4301–4308.

[11] A. Pervez and D. Lee, "Learning task-parameterized dynamic movement primitives using mixture of gmms," *Intelligent Service Robotics*, vol. 11, no. 1, pp. 61–78, 2018.

[12] Y. Huang, J. Silvério, L. Rozo, and D. G. Caldwell, "Generalized task-parameterized skill learning," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 5667–5474.

[13] A. Sena, B. Michael, and M. Howard, "Improving task-parameterised movement learning generalisation with frame-weighted trajectory generation," *arXiv preprint arXiv:1903.01240*, 2019.

[14] T. Giorgino *et al.*, "Computing and visualizing dynamic time warping alignments in r: the dtw package," *Journal of statistical Software*, vol. 31, no. 7, pp. 1–24, 2009.

[15] Y. Grandvalet, S. Canu, and S. Boucheron, "Noise injection: Theoretical prospects," *Neural Computation*, vol. 9, no. 5, pp. 1093–1108, 1997.

[16] K. Matsuoka, "Noise injection into inputs in back-propagation learning," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 3, pp. 436–440, 1992.

[17] Z. He, A. S. Rakin, and D. Fan, "Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 588–597.

[18] Z. Zhao, C. Gan, J. Wu, X. Guo, and J. Tenenbaum, "Augmenting policy learning with routines discovered from a single demonstration," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, 2021, pp. 11 024–11 032.

[19] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg, "Dart: Noise injection for robust imitation learning," in *Conference on robot learning*. PMLR, 2017, pp. 143–156.

[20] D. S. Brown, W. Goo, and S. Niekum, "Better-than-demonstrator imitation learning via automatically-ranked demonstrations," in *Conference on robot learning*. PMLR, 2020, pp. 330–359.

[21] S. Li, N. Figueroa, A. Shah, and J. A. Shah, "Provably safe and efficient motion planning with uncertain human dynamics."

[22] F. Zhang, A. Cully, and Y. Demiris, "Probabilistic real-time user posture tracking for personalized robot-assisted dressing," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 873–888, 2019.

[23] A. Clegg, Z. Erickson, P. Grady, G. Turk, C. C. Kemp, and C. K. Liu, "Learning to collaborate from simulation for robot-assisted dressing," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2746–2753, 2020.

[24] R. P. Joshi, N. Koganti, and T. Shibata, "A framework for robotic clothing assistance by imitation learning," *Advanced Robotics*, vol. 33, no. 22, pp. 1156–1174, 2019.

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

[26] J. Hoyos, F. Prieto, G. Alenyà, and C. Torras, "Incremental learning of skills in a task-parameterized gaussian mixture model," *Journal of Intelligent & Robotic Systems*, vol. 82, no. 1, pp. 81–99, 2016.