Learning Task-Parameterized Skills from Few Demonstrations

Anonymous Author(s)

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 the method less applicable to real-world problems. Therefore, training with fewer demonstrations/situations is desirable. In this paper, we utilize task parameters to generate new demonstration data that augments the original training dataset for policy improvements, thus allowing learning task-parameterized skills with few demonstrations.

Keywords: Learning from Demonstration, Task-Parameterized Learning, Assistive Robots

1 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 model. 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 ambiguity and consequently compromise the policy [3].

The contribution of this paper is a concept for learning task-parameterized skills from few demonstrations. Instead of solely imitating the expert, it allows generation of new demonstration data that aggregate the original dataset for improving the TP-GMM model. 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 algorithm is presented in Sect. 3. In Sect. 4, we describe our framework. Sect. 5 validates our algorithm on a robotic dressing assistance task. Finally in Sect. 6, we conclude.

2 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, Dragan et al. [9] improves original DMP by selecting a Hilbert norm that reduces the deformation in the retrieved trajectory. In more recent research, Zhou et al. [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 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, Pervez and Lee [11] presents an approach that resolves the divergence problem usually associated with GMM type models. Huang et al. [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.

3 Preliminaries

Below we briefly present the TP-GMM algorithm, 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.

3.1 Model Training:

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

$$\{A_n, b_n\}_{n=1}^N$$ (1)

with $A_n, b_n$ represents the orientation and displacement of $n^{th}$ reference frame.

Given $M$ demonstrated trajectory $\{\xi_m\}_{m=1}^M$ for the $m^{th}$ demonstration, we assign the corresponding $N$ reference frames: $\{A_m, n, b_m, n\}_{n=1}^N$ 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 defined by:

$$\{\pi_k, \mu_k^n, \Sigma_k^n\}_{n=1}^N_{k=1}^K$$ (2)
where $\pi_k$ is the $k^{th}$ mixing coefficient, and $\mu^n_k$, $\Sigma^n_k$ are respectively the center and covariance matrix of the $k^{th}$ Gaussian component in frame $n$.

### 3.2 Fusion for New Situations

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

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

a GMM that produce the motion in the new situation can be derived using (2) and (3) in two steps.

**Step 1.** for $n^{th}$ new reference frame, we transform the Gaussian distributions in $n^{th}$ frame in (2) into the new frame using (3):

$$\hat{\mu}^n_k = \hat{A}_n \mu^n_k + \hat{b}_n,$$

$$\hat{\Sigma}^n_k = \hat{A}_n \Sigma^n_k \hat{A}_n^T$$

We apply (4) to all $K$ components in (2).

**Step 2.** for every component, we fuse the transformed Gaussian distributions in $N$ frames (which we obtained from (4) in step 1) by:

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

The new GMM that adapt to the new frames is then $\{ \pi_k, \hat{\mu}_k, \hat{\Sigma}_k \}_{k=1}^K$. The new GMM can be used for motion generation in the new situation via GMR.

### 4 Methods

TP-GMM achieve good generalization performance by assigning reference frames to describe the task. Nevertheless, for retrieving the relevant contextual information in different frames, multiple demonstrations need to be collected in different situations. Creating different situations usually indicate moving objects around in the environment with both translations and rotations, which is a tedious process and sometimes impractical. It also increases the risk of ambiguity in demonstrations. Therefore, it is desirable to reduce the amount of demonstrations needed for training.

#### 4.1 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 for assigning the relevant frame to various parts of the movement in order to obtain a good policy in different situations.

In the model training step presented in Sect. 3.1, the learned TP-GMM in the form of (2) maximizes the 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.

Since task parameters are positions and orientations of reference frames, we can automatically generate new situations and subsequently new demonstration data with a learned TP-GMM model. By defining a suitable selection criteria, feasible data can be selected to augment the training dataset to better understand the data distribution in each reference frame, and subsequently improve the policy. A better policy generation can be reflected by reduction of the difference between the reproduced and initial demonstrated policy. Therefore, we use this difference as the cost to select feasible data to add to the training dataset.

We’d like to mention that 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. Selecting feasible data with the cost and adding the data to the training dataset helps to better understand the distributions in each frame, rather than reinforcing the original policy.
4.2 Learning Task Parameterized Skills from Few Demonstrations

We design Alg. 1 under the premise for learning task-parameterized skill with few demonstrations. We explain the steps in the algorithm in more details below.

Algorithm 1 Learning Task-Parameterized Skills from Few Demonstrations

Demonstration Collection: Initial \( \mu \) demonstrations of the task in distinctive situations: \( D_{\text{init}} \).

Cost Computation: \( \mathcal{J} = f(\mathcal{P}, D_{\text{init}}) \) \( \text{iter} = 0, \ n_d = \mu, \ D = D_{\text{init}} \)

while \( n_d \leq M \) and \( \text{iter} < L \) do

Generate a New Situation

Apply \( \mathcal{P} \) in the new situation and produce new motion data \( D_n \) in that situation

Aggregate datasets: \( D' \leftarrow D \cap D_n \)

Retrain the TP-GMM \( \mathcal{P}' \) from \( D' \)

Compute the cost with new TP-GMM \( \mathcal{J}' = f(\mathcal{P}', D_{\text{init}}) \)

if \( \mathcal{J}' < \mathcal{J} \) then

\( D = D' \)

\( \mathcal{P} = \mathcal{P}' \)

\( n_d = n_d + 1 \)

\( \text{iter} = \text{iter} + 1 \)

else

\( \text{iter} = \text{iter} + 1 \)

end if

end while

return \( \mathcal{P} \)

Demonstration Collection: In this step, we collect \( \mu \) number of demonstrations (where \( \mu \geq 2 \)) for the initial training of the TP-GMM model. In order to avoid that the final policy over-fits to some local configurations, the demonstrations should be collected in distinctive situations. Since in TP-GMM, the situations are described with reference frames, the distinctive measure can be the distances between the corresponding reference frames, and the angles that represents difference in orientations.

Training TP-GMM from initial demonstrations is a standard procedure described in Sect. 3.

Cost Computation: 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 [14]. For \( \mu \) number of initial demonstrations (each with a distinctive situation), we represent the cost as:

\[
\mathcal{J} = \frac{1}{\mu} \sum_{i=1}^{\mu} \text{DTW. normalized_distance}(y(i), \xi_d(i))
\]  

(6)

where \( y(i) \) is the reproduction of initial demonstrations from the TP-GMM and \( \xi_d(i) \) is the initial expert demonstration in the \( i^{th} \) instance. The cost is used for deciding whether we should update the dataset and TP-GMM or not.

Generate a New Situation: In this step, we generate random reference frames that ideally lie the convex regime of any two references frames of the initial \( \mu \) demonstrations. Providing the demonstrations, each associated with \( N \) frames we have:

\[
\{\{A_{j,n}, b_{j,n}\}_{n=1}^{N}\}_{j=1}^{\mu}.
\]

(7)

To generate a new situation, we first randomly select two demonstrations within all initial demonstrations: \( \{A_{a,n}, b_{a,n}\}_{n=1}^{N}, \{A_{b,n}, b_{b,n}\}_{n=1}^{N} \) where \( 1 \leq a, b \leq \mu \land a \neq b \). Subsequently, we compute a random transformation \( \{A_{\text{new},n}, b_{\text{new},n}\}_{n=1}^{N} \) that lies in between the two frames, and satisfies all task constraints such as kinematic limits.

Once we have the new situation, a new motion data \( D_n \) can be obtained from the TP-GMM. We then augment the training dataset with the new 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 new situations generation step. Note that the cost is only calculated based on the expert demonstrations, and does not include the algorithm generated demonstrations.

The algorithm has two termination conditions. The maximum number of the demonstrations in the training dataset $D$ denoted by $M^1$ 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.

5 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. Zhang et al. [15] uses a hybrid force/position control with simple planning for dressing, while Clegg et al. [16] use deep reinforcement learning (DRL) to simultaneously train human and robot control policies as separate neural networks using physics simulations. Although DRL yields satisfactory dressing policies, applying DRL in 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. Joshi et al. [17] 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. Pignat and Calinon [18] 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 [19], 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 model 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 $p_{sh}$, elbow $p_{el}$ and wrist $p_{wr}$ 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:

$$\{A_{sh}, b_{sh}\}, \{A_{wr}, b_{wr}\}, \quad (8)$$

where $b_{sh} = p_{sh}$, $b_{wr} = p_{wr}$. For two orientations $A_{sh}$ and $A_{wr}$, the $x$ axis is defined parallel to vector $p_{sh}p_{el}$ and $p_{wr}p_{el}$ respectively. Figure 1c shows two postures and their respective references frames at the shoulder and wrist.

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:

$$\hat{\mathbf{p}} = \left[ \begin{array}{c} p \\ \delta p \end{array} \right] \in \mathbb{R}^6, \quad \hat{\mathbf{A}} = \left[ \begin{array}{cc} A & 0 \\ 0 & \Lambda \end{array} \right] \in \mathbb{R}^{6 \times 6}, \quad \hat{\mathbf{b}} = \left[ \begin{array}{c} b \\ 0 \end{array} \right] \in \mathbb{R}^6 \quad (9)$$

Following the steps described in Sect. 3.1, we can obtain a TP-GMM model in the form of (2).

The dressing motion is generated by integrating the output of the GMR conditioned on the current position.

\(^1\)Note that $M > \mu$ as in the initialization, we have already $\mu$ demonstrations
Figure 1: Two demonstrations of the dressing task on two distinctive postures. Figure (c) shows two arm postures in (a) and (b) by the dashed lines. Demonstrated path are scattered points in red and blue.

We collect two demonstrations of the dressing task on two distinctive postures respectively (Fig. 1). 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. Rows in Fig. 2 show progressively new demonstration data being added by algorithm and resulting changes of the GMM in each frames. The data colored in blue are initial expert demonstrations and red are algorithm generated demonstrations. The first two columns are $3D$ data in the shoulder frame plotted with $xy$ and $xz$, third and fourth are the data in the wrist frame. In the figure and subsequent instances, we denote different TP-GMM by $x + y$, where $x$ is the number of expert demonstrations and $y$ is the number of algorithm generated demonstrations for training the TP-GMM.

Table 1 shows the reduction of the cost defined in (6), after each new generated demonstration data add to the training dataset. Generated demonstrations 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.

<table>
<thead>
<tr>
<th>Number of Demonstrations (expert + generated)</th>
<th>2 + 0 (init)</th>
<th>2 + 1</th>
<th>2 + 2</th>
<th>2 + 3</th>
<th>2 + 4</th>
<th>2 + 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>0.0826</td>
<td>0.0407</td>
<td>0.0244</td>
<td>0.0242</td>
<td>0.0203</td>
<td>0.0185</td>
</tr>
</tbody>
</table>

We test the policy from the initial (2 expert demonstrations + 0 generated demonstrations), an intermediate (2 expert demonstrations + additional 3 generated demonstrations) and final TP-GMM (2 expert demonstrations + additional 5 generated demonstrations) obtained from our algorithm. One TP-GMM trained from 7 expert demonstrations (including the 2 we used to initialize TP-GMM for our algorithm) serve as a baseline for comparison. These TP-GMM are tested on 5 different postures. We define 2 success condition for checking the success rate of policy generated from TP-GMM:

1. offline success
2. experiment success

If a trajectory is able to reach above and around shoulder, we consider the policy can dress the arm successfully offline. However, this condition does not necessarily imply success dressing in the real robotics experiment: i.e., the armscye can get stuck at the elbow or the robot may hit the arm during trajectory execution. Therefore, an additional measure, experiment success, measures if the trajectory is executable in practice. The trajectory and posture are presented in Fig. 3. The success rate of task execution is summarized in Tab 2. Our algorithm improves the success rate of original TP-GMM and outperforms the baseline TP-GMM that is trained with the same number of expert demonstrations in the experiments.
Figure 2: The new demonstration data being added by algorithm and resulting changes of the GMM in each frames.

Figure 3: Dressing trajectories on 5 different postures by different TP-GMM. The first row shows the setup and the postures in the experiment, the second row show the corresponding trajectory obtained from different TP-GMM on the setup above. Red: TP-GMM 2 + 0, Green: TP-GMM 2 + 3, Blue: TP-GMM 2 + 5 (The final resulting TP-GMM), Pink: TP-GMM 0 + 7 (baseline with 7 expert demonstrations).
Table 2: Task success rate on 5 postures for different TP-GMM.

<table>
<thead>
<tr>
<th>TP-GMM</th>
<th>2 + 0</th>
<th>2 + 3</th>
<th>2 + 5</th>
<th>7 + 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offline success rate</td>
<td>20%</td>
<td>60%</td>
<td>100%</td>
<td>80%</td>
</tr>
<tr>
<td>Robotic experiments success rate</td>
<td>20%</td>
<td>40%</td>
<td>80%</td>
<td>60%</td>
</tr>
</tbody>
</table>

6 Conclusion

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 on real robot experiments with a dressing assistance task.

It is noteworthy that the algorithm might also be used for improving the learned TP-GMM with arbitrary demonstrations. However, if the demonstrations already capture the distributions in each reference frame well enough, the improvement might be marginal.

References


