Learning Task-Parameterized Skills from Few Demonstrations

Jihong Zhu, Michael Gienger, and Jens Kober

Abstract—Moving away from repetitive tasks, robots nowadays demand versatile skills that adapt to different situations. Task-parameterized learning improves the generalization of motion policies 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 comprehensively create different situations is non-trivial thus renders the method less applicable to real-world problems. Therefore, training with fewer demonstrations/situations is desirable. This paper presents a novel concept to augment the original training dataset with synthetic data for policy improvements, thus allows learning task-parameterized skills with few demonstrations. Videos of the experiments are available at https://sites.google.com/view/tp-gmm-from-few-demos/

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 training 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 bringing in 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 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], Gaussian mixture models (GMMs) [6] and more recently conditional neural movement primitives [7] 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 [8], [9]. LfD benefits from demonstration data for skill learning, nevertheless, it is also limited by the reliance on data [1]. Improvement can be made consider data efficiency or policy generalization.

For improving data efficiency in LfD: meta-imitation learning reuses data collected for different tasks [10] thus boosts data efficiency. Another common approach for improving efficiency is synthetic data augmentation. The authors of [11] show that policy improvement can also be made from a single demonstration that is augmented with noise. In robotics LfD, DART is introduced for robust imitation learning [12]. Recently, the authors of [13] present a method that adds different level of noise into the demonstration data for improving the reward learning in an inverse reinforcement learning setting.
For improving generalization in LfD: \cite{14} improves original DMP by selecting a Hilbert norm that reduces the deformation in the retrieved trajectory. In a more recent research, \cite{15} combines 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) \cite{2}, \cite{16}, \cite{17}. By combining the movement model with task parameters, TP-GMM(HMM) can produce a policy that adapts to different situations.

**Improvements on TP-GMM:** Multiple improvements on the original TP-GMM have been made in previous research. By learning the forcing term in DMP with TP-GMM, \cite{16} presents an approach that resolves the divergence problem usually associated with GMM type models. \cite{17} introduces weighting to TP-GMM and \cite{18} build on the idea and proposes to infer weights from co-variances in the normal distribution. While the focus of above-mentioned improvements is on better generalization of the policy. In these approaches, the distributions well in each frame. Subsequently, the model would fail to obtain a decent fusion for policy generation.

**Model Training:**

TP-GMM to new situations. In robotic, depending on the task, \cite{2} introduces the term in DMP with TP-GMM, \cite{16} presents an approach that resolves the divergence problem usually associated with GMM type models. \cite{17} introduces weighting to TP-GMM and \cite{18} build on the idea and proposes to infer weights from co-variances in the normal distribution. While the focus of above-mentioned improvements is on better generalization of the policy. In these approaches, the distributions well in each frame. Subsequently, the model would fail to obtain a decent fusion for policy generation.

**Fusion for New Situations**

For a new situation defined by a set of \( N \) references frames

\[
\{ \mathbf{A}_n, \mathbf{b}_n \}_{n=1}^{N},
\]

a GMM that produces 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):

\[
\hat{\mu}_k = \mathbf{A}_n \mu_k + \mathbf{b}_n, \quad \hat{\Sigma}_k = \mathbf{A}_n \Sigma_k \mathbf{A}_n^T
\]

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:

\[
\Sigma_k^{-1} = \sum_{n=1}^{N} \Sigma_k^{-1}, \quad \hat{\mu}_k = \sum_{n=1}^{N} \Sigma_k^{-1} \mu_n
\]

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

\[
\mu_k = \begin{bmatrix} \mu_k^T \\ \Sigma_k \end{bmatrix}, \quad \Sigma_k = \begin{bmatrix} \Sigma_k^{\text{FO}} & \Sigma_k^{\text{OT}} \\ \Sigma_k^{\text{OT}} & \Sigma_k \end{bmatrix}
\]

Given input data \( y^T \) (current time instance for time-based TP-GMM or positions for trajectory-based TP-GMM), the motion generation in the new situation is solved using Gaussian Mixture Regression (GMR) \cite{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. Fig. 1 presents the comparison between a GMM and TP-GMM. While the former explicitly maximize the likelihood of motion data in one situation, the latter learns the data representation in each frames under different situations.

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

**B. Fusion for New Situations**

For a new situation defined by a set of \( N \) references frames

\[
\{ \mathbf{A}_n, \mathbf{b}_n \}_{n=1}^{N},
\]

a GMM that produces 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):

\[
\hat{\mu}_k = \mathbf{A}_n \mu_k + \mathbf{b}_n, \quad \hat{\Sigma}_k = \mathbf{A}_n \Sigma_k \mathbf{A}_n^T
\]

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:

\[
\Sigma_k^{-1} = \sum_{n=1}^{N} \Sigma_k^{-1}, \quad \hat{\mu}_k = \sum_{n=1}^{N} \Sigma_k^{-1} \mu_n
\]

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

\[
\mu_k = \begin{bmatrix} \mu_k^T \\ \Sigma_k \end{bmatrix}, \quad \Sigma_k = \begin{bmatrix} \Sigma_k^{\text{FO}} & \Sigma_k^{\text{OT}} \\ \Sigma_k^{\text{OT}} & \Sigma_k \end{bmatrix}
\]

Given input data \( y^T \) (current time instance for time-based TP-GMM or positions for trajectory-based TP-GMM), the motion generation in the new situation is solved using Gaussian Mixture Regression (GMR) \cite{2}.
We elaborate on the steps marked with **bold – italic** in this section.

**Inputs:** In this step, we mainly discuss the demonstrations collection. We collect \( \mu \) demonstrations (where \( \mu \geq 2 \)) \( \mathcal{D}_{\text{init}} \) for the initial training of the TP-GMM. The number \( \mu \) depends on the complexity of the task (i.e., tasks with more frames and less constraints may require a larger \( \mu \)). 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 difference in Euler 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 \( \mathcal{D}_{\text{init}} \) and the TP-GMM \( \mathcal{P} \) as inputs.

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

\[
J_{\text{RMS}} = \frac{1}{\mu} \sum_{i=1}^{\mu} \sqrt{\| y_i - \xi_i \|},
\]

where \( y_i \) is the reproduction of initial demonstrations from the TP-GMM and \( \xi_i \) is the initial expert demonstration, both in the \( i^{\text{th}} \) 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.

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

\[
\phi(K) = (\phi_y(K), \phi_\xi(K)),
\]

where \( \phi_y(K) \in y \), and \( \phi_\xi(K) \in \xi \). The warping function maps the data in \( y \) to the data in \( \xi \) 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 \( y \) and \( \xi \):

\[
d_\phi(y, \xi) = \frac{1}{T} \sum_{K=1}^{T} d(\phi_y(K), \phi_\xi(K))\zeta_\phi(K)/Z,
\]

where \( \zeta_\phi(K) \) is a weighting coefficient and \( Z \) is the normalization constant (we refer readers to [19] for a detailed definition). The warping function is computed by minimizing the cost defined in (8):

\[
D(y, \xi) = \min_\phi d_\phi(y, \xi)
\]

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

\[
J_{\text{DTW}} = \frac{1}{\mu} \sum_{i=1}^{\mu} D(y(i), \xi_d(i))
\]

We choose the cost depending on the type of TP-GMM and use it as a selection criterion 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:
In this section, we analyze data generation methods proposed in Sect. IV for Alg. 1 in terms of cost reduction and policy generalization on a simulated task\(^2\). 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. 2a. The shape imposes constraints for the motion. We use six expert demonstrations, and split them into a training and validation set (Fig. 3a). Each contains three expert demonstrations, and we observe that the outputs is the final TP-GMM that has the lowest cost after the dataset augmentation.

\(^2\)The simulation results in this section is coded based on pbdlib: https://gitlab.idiap.ch/rli/pbdlib-matlab/
which contains three new expert demonstrations in new situations\(^2\).

For each selection (criterion), we improve the initial TP-GMM models using three methods: namely Noise, RF and RF+Noise. The termination condition for the algorithm is set to be \( M = 8 \) (max number of demonstrations) and \( L = 50 \) (max 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 save the models with the maximum cost reduction.

We hence get 6 improved TP-GMM models. We then compute the cost that defined in (7) for the training and validation set for each improved model. While the training cost indicates the performance of policy in reproducing the expert demonstrations, the validation cost measures the policy generalization in new situations. The results are summarized in Tab. I. We use the cost of the initial TP-GMM as the reference quantity and provides a percentage value besides each cost value. The increased cost is marked with red and decreased with cyan in the table.

From Tab. I we observe that in the original selection, all three data augmentation methods can reduce the cost on the training set. As the cost is reduced, they all show better performance in terms of reproduction of the training set as also shown in Fig. 4. As for the generalization selection, since the criterion for selecting synthetic data is defined with regard to the validation set, the cost on training set remain close to the reference in Noise and RF+Noise, and is even higher in the case of RF. For the increase cost, we suspect that adding more trajectories with the generalization selection is likely to make the final model over-fit to the validation set, thus the cost on the training set increases. However, due to the page limit, we cannot discuss this in detail here.

With the original selection, Noise is able to reduce the cost the most. It is reasonable as the augmented data is the demonstrations from the training set plus white noise.

\(^2\)Note that this selection criterion is only used as an ablation for the best-case performance in the new situations. In practice it requires more demonstrations than the original selection criterion.

Reduction of the training cost is one indicator of the performance. More importantly, generalization to new situations is the core capability of TP-GMM. In that regard, Noise performed poorly on the validation set with an increased cost, indicating overfitting to the training set. The phenomena can be visually confirmed in Fig. 5c (black trajectories) where Noise turns out to magnify some unwanted movements in the produced motions in the new situation.

RF and RF+noise, on the other hand, provide a reasonable cost reduction on both training and validation sets. Such reduction is manifested in the improved motions in Fig. 5 as compared to the initial TP-GMM in Fig. 3(b) – (d). Although the cost reduction on the validation set is slightly higher than in the generalization selection, but they are comparable. This point is further illustrated in Fig. 5 which shows that the motions produced from RF and RF+noise in the original selection is very similar to RF+Noise in the generalization selection.

Another way of cost reduction without generating new data is to increase the number of Gaussian \( K \) in (2) 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 demonstrations, and has the possibility to magnify unwanted movements in the demonstration similar to Noise. One good indicator for checking the fitting is the reconstruction error of demonstrations we defined in (7) and (10). For selecting a reasonable \( K \), as suggested in [2], methods such as Bayesian information criterion [23] and Dirichlet process [24] can be used.

In a nutshell, by explicitly using reproduction error as the selection criterion, all three data generation methods are able to generate synthetic motions for data augmentation to improve TP-GMM. While TP-GMM produced from Noise tends to over-fit to original demonstrations and may magnify unwanted movements, the TP-GMM produced from RF and RF+Noise are improved over the original TP-GMM in reproduction and also generalization. The generalization is comparable with

### TABLE I

|                      | Original Selection | Generalization Selection | Init. TP-GMM |
|----------------------|--------------------|--------------------------|-------------|
|                      | Noise   | RF     | RF+Noise | Noise   | RF     | RF+Noise | Noise   | RF     | RF+Noise | Noise   | RF     | RF+Noise |
| training cost        | 1.97 (91%) | 2.08 (96%) | 2.04 (94%) | 2.21 (102%) | 2.41 (111%) | 2.17 (100%) | 2.17 (100%) | 2.17 (100%) | 2.17 (100%) |
| validation cost      | 2.50 (118%) | 1.85 (88%) | 2.08 (87%) | 1.48 (70%) | 1.79 (85%) | 1.82 (86%) | 2.11 (100%) | 2.04 (94%) | 2.08 (96%) | 2.08 (96%) |

---

Fig. 4. Reproduction of expert demonstrations in training set with original selection, the expert demonstrations are in blue and reproduction in red.

Fig. 5. Generalization to validation set by TP-GMM produced with Noise (black), TP-GMM produced with RF (green), TP-GMM produced with RF+noise (cyan), all using the original selection criterion and TP-GMM produced with RF+Noise using the generalization selection criterion (magenta).
a generalization selection where the cost is computed with regard to 3 new expert demonstrations in new situations.

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 armseye. 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 [25] 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 [26], while [27] 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. The dressing policy for a single static posture is encode by a DMP in [28]. When considering multiple different static postures and the policy needs to adapt, task-parameterized approaches becomes more relevant. Sensory information and motor commands are combined in [29] as a joint distribution in a hidden semi-Markov model, and then coupled with a task-parameterized model to generalize to different situations. In [30], the authors propose 3 techniques to incrementally updates the TP-GMM when new demonstration data is available and tested with a dressing assistance scenario. All techniques in [30] require new expert demonstrations and focus on how to update the TP-GMM while our method focuses on how to generate synthetic data for policy improvement for TP-GMM. Both [29], [30] require several expert demonstrations (either at the beginning or incrementally) for generalizing the dressing task. In this section, we demonstrate that using our framework, the robot can learn to dress with only two expert demonstrations.

The dressing trajectory needs to adapt to different arm postures. The postures are described with positions of shoulder \( p_{\text{sh}} \), elbow \( p_{\text{el}} \) and wrist \( p_{\text{wr}} \) 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:

\[
\{ A_{\text{sh}}, b_{\text{sh}} \}, \{ A_{\text{wr}}, b_{\text{wr}} \},
\]

(13)

where \( b_{\text{sh}} = p_{\text{sh}} \), \( b_{\text{wr}} = p_{\text{wr}} \). For two orientations \( A_{\text{sh}} \) and \( A_{\text{wr}} \), the \( x \) axis is defined parallel to vector \( p_{\text{sh}} p_{\text{el}} \) and \( p_{\text{wr}} p_{\text{el}} \) respectively. Fig. 6 shows different postures and their respective reference frames at the shoulder and wrist with coordinate depicted with red, green and blue (RGB) arrows.

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.

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:

\[
\hat{p} = \left[p \atop \delta p\right] \in \mathbb{R}^{6}, \quad \hat{A} = \left[A \atop 0\right] \in \mathbb{R}^{6 \times 6}, \quad \hat{b} = \left[b \atop 0\right] \in \mathbb{R}^{6}
\]

(14)

Following the steps described in Sect. III-A, we can obtain a TP-GMM in the form of (2). 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, the demonstration dataset contains human demonstrations in two different postures. Fig. 6 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 RF and RF+noise as the data generation methods. In RF+noise the SNR is set to be 30 decibels. The cost reduction is indicate in Fig. 7. Two cases have different initial cost as the demonstration data are different.

In addition, likewise in Sect. V, we test the generalization
of the original and the final obtained TP-GMM. These TP-GMM are tested on 5 different postures. We define the success condition as follows. 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 from initial and final (synthetic data augmented) TP-GMM is summarized and compared with the baseline in Table II.

We observe that Case A improves more than Case B in terms of generalization. Take a closer look at the initial demonstrations of both cases, we observe that compared with Case A, in Case B, the initial two postures are more similar to each other and so are the resulting initial expert demonstrations. This is also consistent with Fig. 7 that the initial cost of Case B is less than Case A. The TP-GMM in Case B is probably over-fit to the data in the region which might explains Case A outperforming B in terms of generalization. The best case scenario is able to match the baseline in terms of success rate.

The average run-time in experiment setup for our algorithm is around 20 seconds on a Linux-based i7-9750 PC, which is far less than the efforts and time of doing one extra expert demonstration.

We notice even the baseline did not achieve 100% in success rate. The failure case was stuck at the elbow. The reason might be that the TP-GMM we used only considers reference frames as task-parameter while the dressing path is dependent on additional latent variables. However, finding these variables is beyond the scope of this paper.

We use an example case with the best experiment performance (Case A with RF as the data generation method) to show 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 column. Note that row-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 the algorithm and the resulting changes of the GMM in each frame. The data colored in yellow are initial expert demonstrations and red are synthetic demonstrations. The first two rows are 3D position data in the shoulder frame plotted with $xy$ and $xz$, the third and fourth are the data in the wrist frame. The ellipse represents the co-variance of each multivariate Gaussian.

We run the the algorithm 20 times to obtain the average cost and the number of synthetic trajectories needed for augmenting $n (n = 1 \ldots 5)$ number of new synthetic demonstrations.

$^4$On average collecting one dressing demonstration on our current setup requires around 10 seconds, without considering the time to change posture, and efforts in making the demonstration.

![Fig. 8. The synthetic demonstration data being added by algorithm and resulting changes of the GMM in each frame. The points in yellow and red are respectively original and synthetic demonstrations seen from 2D frames.](image)

The results are presented in Table III. 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.

### Conclusion and Future Works

Task-parameterized learning often requires 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 methods. Although noise injection provides the highest cost reduction, it has the possibility of magnifying 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, distinctive initial demonstrations may contribute to better generalization performance of the algorithm.

TP-GMM needs a well-defined task space with reference frames which limits its application. For the dressing task, all TP-GMMs fail to dress at least one posture. That leads to a limitation of the current method: the exploration is done entirely in the synthetic domain which means the algorithm is not designed to handle environment feedback and can not explicitly use reward signals. As a consequence, the resulting TP-GMM will not able to resolve on the corner cases. RL-based policy search maybe promising in resolving corner cases as reported in ACNMP framework in [31] and adaptive ProMP in [32]. However, for dressing tasks, the reward needs to be carefully designed as reflected in [33]. Alternatively, instead of discarding the trajectories, efforts could be made to exploit them [34] to further enhance the data efficiency.

The current method considers a fix number of Gaussians. As the training data set is augmented with the algorithm, simultaneously increasing the number of Gaussian will probably result in a better approximation of data distribution in each frame and subsequently improve the policy even further.

REFERENCES

[1] H. Ravichandar, A. S. Polyzoros, S. Chernova, and A. Billard, “Recent advances in robot learning from demonstration,” Annual Review of Control, Robotics, and Autonomous Systems, 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 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,” NeurIPS, 2013.

[6] M. Mühlig, M. Gienger, S. Hellbach, J. J. Steil, and C. Goeurck, “Task-level imitation learning using variance-based movement optimization,” in IEEE Int. Conf. on Rob. and Auto., 2009.

[7] M. Y. Seker, M. Imre, H. J. Piater, and E. Uzur, “Conditional neural movement primitives.” in RSS, vol. 10, 2019.

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

[9] A. Nair, B. McGrew, M. Andrzejowicz, W. Zaremba, and P. Abbeel, “Overcoming exploration in reinforcement learning with demonstrations,” in 2018 IEEE Int. Conf. on Rob. and Auto.

[10] D.-A. Huang, S. Nair, D. Xu, Y. Zhu, A. Garg, L. Fei-Fei, S. Savarese, and J. C. Niebles, “Neural task graphs: Generalizing to unseen tasks from a single video demonstration,” in CVPR, 2019.

[11] Z. Zhao, C. Gan, J. Wu, X. Guo, and J. Tenenbaum, “Augmenting policy learning with routines discovered from a single demonstration,” in AAAI, 2021.

[12] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg, “Dart: Noise injection for robust imitation learning,” in CoRL, 2017.

[13] D. S. Brown, W. Goo, and S. Niekum, “Better-than-demonstrator imitation learning via automatically-ranked demonstrations,” in CoRL, 2020.

[14] Z. Zhu, D. Dragan, K. Muelling, J. A. Bagnell, and S. Srinivasa, “Movement primitives via optimization,” in 2015 IEEE Int. Conf. on Rob. and Auto. IEEE, 2015, pp. 2339–2346.

[15] Y. Zhou, J. Gao, and T. Asfour, “Learning via-point movement primitives with inter-and extrapolation capabilities,” in 2019 IEEE/RSJ Int. Conf. on Intell. Rob. and Sys. IEEE, 2019, pp. 4301–4308.

[16] A. Perez and D. Lee, “Learning task-parameterized dynamic movement primitives using mixture of normals,” Intelligent Service Robotics, vol. 11, no. 1, pp. 61–78, 2018.

[17] Y. Huang, J. Silvérto, L. Rozo, and D. G. Caldwell, “Generalized task-parameterized skill learning,” in 2018 IEEE Int. Conf. on Rob. and Auto. IEEE, 2018, pp. 5667–5474.

[18] A. Sena, B. Michael, and M. Howard, “Improving task-parameterised movement learning generalisation with frame-weighted trajectory generation,” in 2019 IEEE/RSJ Int. Conf. on Intell. Rob. and Sys. IEEE, 2019, pp. 4281–4287.

[19] T. Giorgino et al., “Computing and visualizing dynamic time warping alignments in R: the DTW package,” Journal of statistical Software, 2009.

[20] Y. Grandvalet, S. Canu, and S. Boucheron, “Noise injection: Theoretical prospects,” Neural Computation, 1997.

[21] K. Matsuoaka, “Noise injection into inputs in back-propagation learning,” IEEE Transactions on Systems, Man, and Cybernetics, 1992.

[22] Z. He, A. S. Rakin, and D. Fan, “Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack,” in CVPR, 2019, pp. 588–597.

[23] G. Schwarz, “Estimating the dimension of a model,” The annals of statistics, 1978.

[24] S. P. Chatzis, D. Korkinof, and Y. Demiris, “A nonparametric bayesian approach toward robot learning by demonstration,” Robotics and Autonomous Systems, 2012.

[25] S. Li, N. Figueroa, A. Shah, and J. A. Shah, “Provably safe and efficient motion planning with uncertain human dynamics,” in RSS, 2021.

[26] F. Zhang, A. Cully, and Y. Demiris, “Probabilistic real-time user posture tracking for personalized robot-assisted dressing,” IEEE Trans. on Rob., vol. 35, no. 4, pp. 873–888, 2019.

[27] 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 Rob. and Auto. Letters, vol. 5, no. 2, pp. 2746–2753, 2020.

[28] R. P. Joshi, N. Koganti, and T. Shibata, “A framework for robotic clothing assistance by imitation learning.” Advanced Robotics, 2019.

[29] E. Pignat and S. Calinon, “Learning adaptive dressing assistance from demonstration,” Rob. and Auto. Sys., vol. 93, pp. 61–75, 2017.

[30] 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, 2019.

[31] M. Akbulut, E. Oztop, M. Y. Seker, X. Hh, A. Tekden, and E. Ugur, “Acnmp: Skill transfer and task extrapolation through learning from demonstration and reinforcement learning via representation sharing,” in CoRL, 2021.

[32] S. Stark, J. Peters, and E. Rueckert, “Experience reuse with probabilistic movement primitives,” in IEEE/RSJ Int. Conf. on Intell. Rob. and Sys., 2021.

[33] A. Clegg, W. Yu, J. Tan, C. K. Liu, and G. Turk, “Learning to dress: Synthesizing human dressing motion via deep reinforcement learning,” ACM Trans. on Graphics, vol. 37, no. 6, pp. 1–10, 2018.

[34] M. T. Akbulut, U. Bozdogan, A. Tekden, and E. Ugur, “Reward conditioned neural movement primitives for population-based variational policy optimization,” in IEEE Int. Conf. on Rob. and Auto., 2021.