Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

Robot Programming by Demonstration with Local Human Correction for Assembly

Zhiqi Cao, Haopeng Hu, Zhilong Zhao and Yunjiang Lou, *Senior Member, IEEE*

***Abstract*— For the sake of boosting the usability of robots in assembly applications, Programming by Demonstration (PbD) has been regarded as a feasible solution to transfer human worker's skills to robots. With one or more human assembly demonstrations, an assembly policy, which serves to generate a robot motion trajectory given a specific target, can be learned and then used to generate a smooth robot assembly trajectory. However, when it comes to complex and subtle assembly tasks that are widely spreading in industries, the success rate of the learned policy is limited by physical constraints introduced by the human demonstrations, the robot and the task. Motivated by this problem, this work proposes a robot PbD method of the human-in-the-loop framework that draws support from a human operator to reinforce the learned assembly policy. The Gaussian Mixture Model is utilized to pre-structure the policy and it is adjusted by requesting the human operator to provide corrections at critical positions, where unexpected collisions occur. A kinesthetic teaching experiment has demonstrated the effectiveness of the proposed method.**

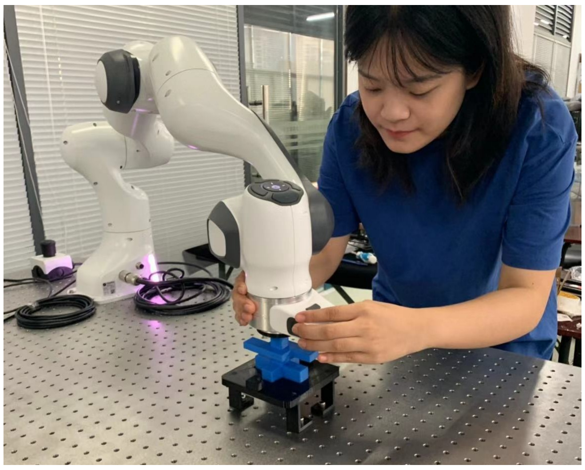


Fig. 1. Kinesthetic teaching for precise assembly

I. Introduction

Nowadays, robots enjoy growing popularity in many fields such as in human-robot collaborative manufacturing. With the increase of labor cost, demand for robots in manufacturing applications, e.g. precise assembly, is rising unprecedentedly. However, when it comes to assembly tasks that require complex assembly skills and movement precision, robot programmers using traditional robot programming codes suffer from complicated and time-consuming programming process. For illustration, the 3C (Computer, Communication and Consumer) assembly line is a representative assembly application where the assembling process is characterized by high precision requirements and complex assembly skills. Worse still, there is frequent change-over in 3C assembly lines. This is not an unique instance but a common issue in the flexible manufacturing field. Frequent re-programming requires a large amount of time and labor which limits the utilization of robots. In addition, most of the workers in 3C assembly lines are non-expert robot users who have deep understanding of the assembly tasks but have little idea of the robot. Those challenges call for a robot programming technique that makes use of human workers' experience as well as avoids complicate professional coding work. Programming by Demonstration (PbD), which also called as Learning from Demonstration and Imitation Learning [7], has drawn increasing interest as a promising approach to tackle all of these challenges mentioned above. It aims to derive a *policy* from the demonstration data provided by the human operator and then generate suitable robot motions given a specific target. In assembly applications, the policy can be regarded as a parametric model able to generate key behaviors that are essential to accomplish the particular assembly task. With a simple demonstration technique such as kinesthetic teaching and teleoperation, programming by demonstration methods enable the non-expert operators to deploy robots in assembly applications at a little cost of labor. When a human operator demonstrates a complex assembly task, he or she mostly makes it with a visual servo method taking visual information from his or her eyes as feedback. Especially at the location where there is a mechanically fit of small clearance, as illustrated by Fig. 1, the closed-loop nature of the control mechanism brings in redundant demonstration motion, i.e. the *overshoot*. As a result, policy derived from those redundant demonstrations may not generate efficient, or even feasible, robot assembly motion. In view of that problem, the human operator can play an active role during the robot assembly process to improve the usability of the policy. With additional help from the operator, some key information of the task can be figured out which helps to reduce the number of demonstrations required to learn a suitable enough policy.

This work proposes a robot programming by demonstration method with local human corrections for assembly tasks. To further argue its practicability, the related robot PbD studies in assembly applications and PbD studies with human

This work was supported partially by the NSFC-Shenzhen Robotics Basic Research Center Program (No. U1713202) and partially by the Shenzhen Science and Technology Program (No.JCYJ20180508152226630).

Cao, Hu, Zhao and Lou (the corresponding author) are with the School of Mechatronics Engineering and Automation, Harbin Institute of Technology, Shenzhen, HIT Campus, University Town of Shenzhen, Xili, Nanshan, Shenzhen, China. louyj@hitsz.edu.cn

evaluative/corrective advice are reviewed next. In literature, pick-and-place is the simplest assembly application studied with PbD framework [1]. There is precision requirement but no *skill* requirement in those works. Furthermore, peg-in-hole problems have been deeply studied as assembly applications in literature. Suarez-Ruiz et al. proposed a kind of dual-arm cooperation assembly system which is based on visual service and force control in order to achieve bimanual pin insertion of a chair [3]. J. S. Laursen et al. applied reversible computing principles to screw a self-tapping bolt into a material by an UR5 robot [4]. In his another work, it proposed a method that makes two union nuts on a tube by a robot [5]. These works' key issue lies in the insertion process but generally assembly tasks may call for complex skills rather than insertion [8]. In view of that problem, M. Kyrarini et al. proposed an adaptation algorithm which can identify the sequence of actions when robots need to perform in new condition[16]. But in precise assembly tasks, detecting critical points from demonstration is challenging problem. The human operator is able to provide active correction/evaluation when the derived policy is working. Luka et al designed a robotic assembly solution by human-in-the-loop teaching method to realize the relative movement of bolts [2]. S. Calinon and A. Billard presented a way to teach a robot identify the gestures by putting the human teacher in the learning loop [14]. Those methods require the operator to perform more demonstrations. Y. Schroecker selected several taught via-points interactively to guide the policy search [10]. In order to transfer human assembly skills to robots for precise assembly applications and inspired by the COACH framework (Corrective Advice Communicated by Humans [9]), this work presents a human-in-the-loop PbD method whose workflow is exhibited in Fig. 2. This method consists of two phases, namely the *unsupervised policy learning phase* (the cream block) and the *policy reinforcement phase* (the azure block). During the former phase, the Gaussian mixture model is utilized to model the assembly policy which is widely used in PbD studies [18]. However, in most cases the policy fails to work well in precise assembly applications. It is partially because the learned policy depends on the sub-optimal demonstrations and no environment model is considered. When unexpected collisions happen during the latter phase, human corrections are requested and utilized to reinforce the assembly policy as it is assumed that humans, such as workers in assembly lines, always have a deep understanding of the assembly task.

The article is organized as follows: Section II introduces the proposed human-assisted programming by demonstration method in detail. As the method can be decomposed into two phases, section II-B and section II-C discusses the operating process of the unsupervised assembly policy learning phase and the assembly policy reinforcement phase respectively after the executing process has been explained in section II-A. An experiment in section III demonstrate the effectiveness of the proposed method by assembling two components. Section IV concludes the article and sheds light on some future research interests.

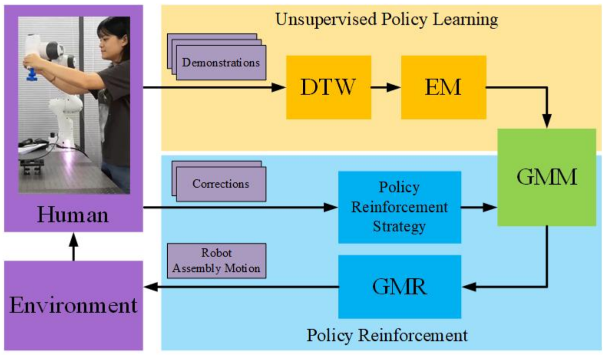


Fig. 2. Flow of the PbD method with human correction

II. PBD WITH LOCAL HUMAN CORRECTION

A. Overview

Given an explicit assembly target, here an assembly skill is defined to be critical movements to carry out the assembly task without any unexpected collisions. As shown in Fig. 2 and algorithm 1, the assembly skill is represented by a Gaussian mixture model (GMM) learned from multiple human demonstrations. Moreover, the GMM model is adjusted by requesting human corrections when the robot performs the same assembly task. In this study, the human operator only provides local corrections which are a user-friendly and labor-saving strategy. The rest of the section discusses the details of the proposed method.

B. Unsupervised Assembly Policy Learning

As a type of probabilistic mixture distribution, GMM is a widely-used model to study the shared features of multiple motion trajectories $\xi = \{\xi^{(1)}, \xi^{(2)}, \ldots, \xi^{(M)}\}$ of the same task in which $\xi^{(i)} \in \mathbb{R}^{D\times N_i}, i = 1, 2, \ldots, M$. D is the dimension of state variable and N_i is the number of states (or elapsed time steps) in the i-th demonstration. In this study, $\xi_n^{(i)} = [x_n^{(i)}, y_n^{(i)}]^T$ in which $x \in [0,1]$ is the *decaying variable* and $y \in \mathbb{R}^{D-1}$ is the position variable in the demonstrated

Algorithm 1 Programming by Demonstration with local human correction method

```
Require: Human kinesthetic assembly demonstrations $\xi$.
 1: Perform DTW: $\bar{\xi} = \mathrm{DTW}(\xi)$.
 2: Initialize GMM parameters $\theta^{(0)}$ by K-Means algorithm.
 3: Learn a GMM model: $\theta^* = \mathrm{EM}(\bar{\xi})$.
 4: $i = 1$
 5: repeat
 6:     Generate an assembly trajectory $\hat{\xi}^{(i)}$ by GMR.
 7:     Control the robot to follow the trajectory $\hat{\xi}^{(i)}$.
 8:     if Unexpected collision occurs then
 9:         Read the decaying step $\tilde{x}^{(i)}$ when collision occurred.
10:         Request human correction demonstration $\tilde{y}^{(i)}$.
11:         Adjust the GMM model.
12:         $i = i + 1$
13:     end if
14: until No unexpected collision occurs.
```

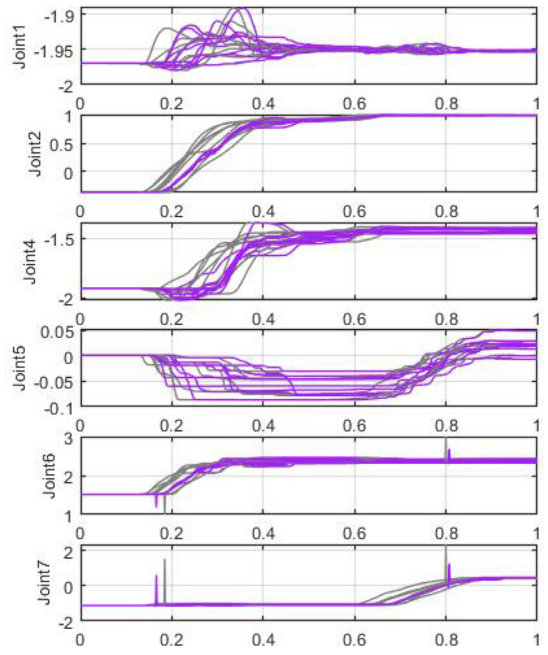


Fig. 3. Demonstrated motion trajectories before and after DTW

trajectories. As discussed in section I, in the practical assembly application, it is irrational to require a human operator perform one assembly task several times within the same duration. Demonstration trajectories of the same task may seem totally different whose temporal sequences are even encoded by a decaying variable $x \in [0,1]$ rather than absolute time series. To improve regression quality, dynamic time wrapping (DTW) [11], as stated in line 1 in algorithm 1, is used here to serve as a template matching process to temporally align the demonstrations ξ to $\bar{\xi}$. By computing the similarity among demonstrations, DTW is widely-used in the literature on applying GMM to PbD problems [12]. Performing DTW over high-dimensional demonstrations will require an amount of computational cost. This is not a problem as in the proposed method, the unsupervised policy learning phase is operated offline and only once. After DTW and by data re-sampling, all the aligned demonstrations share the same number of data N.

Figure 3 shows the demonstrated motion trajectories in joint space before and after DTW process. The grey curves stand for the raw demonstration data while the purple curves stand for those data processed by DTW. Details about those data can be found in section III. After DTW, the Gaussian mixture model is learned by Expectation-Maximization (EM) algorithm. Assume that all of the demonstrations are generated by a mixture of K Gaussian distributions in *latent space*.

$$p(\xi_n) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\xi_n; \mu_k, \Sigma_k) \tag{1}$$

where π_k, which is called *prior*, is defined as the probability of ξ_j generated by distribution k satisfying $\sum_{k=1}^{K} \pi_k = 1$ and

$$\mathcal{N}(\xi_n; \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^D |\Sigma_k|}} e^{-\frac{1}{2}\left((\xi_n - \mu_k)^T \Sigma_k^{-1} (\xi_n - \mu_k)\right)}$$

Note that the hyper-parameter K must be set manually. Larger K will tend to more precise estimation of the latent space, nonetheless, computational cost will increase simultaneously. The optimal number K can be determined w.r.t. optimization objectives such as AIC [19], BIC [20] and DIC [21]. Here the *Bayesian Information Criterion* (BIC) is considered:

$$J_\theta = -\sum_{n=1}^{N} \log(p(\xi_n)) + \frac{n_p}{2} \log(K) \tag{2}$$

n_p is the number of free parameters for a mixture of K Gaussian distributions. Objective (2) can be seen as a tradeoff between maximizing the model's log-likelihood and minimizing the number of parameters needed. Given demonstrations $\bar{\xi}^{(d)}$, parameters $\theta_k : \{\pi_k, \mu_k, \Sigma_k\}$ are estimated iteratively by EM algorithm:

$$\begin{aligned}
\pi_k^{(i+1)} &= \frac{\sum_{n=1}^{N} p_{k,n}^{(i+1)}}{N} \\
\mu_k^{(i+1)} &= \frac{\sum_{n=1}^{N} p_{k,n}^{(i+1)} \xi_n}{\sum_{n=1}^{N} p_{k,n}^{(i+1)}} \\
\Sigma_k^{(i+1)} &= \frac{\sum_{n=1}^{N} p_{k,n}^{(i+1)} (\xi_n - \mu_k^{(i+1)})(\xi_n - \mu_k^{(i+1)})^T}{\sum_{n=1}^{N} p_{k,n}^{(i+1)}}
\end{aligned} \tag{3}$$

in which

$$p_{k.t}^{(i+1)} = \frac{\pi_k \mathcal{N}(\xi_n; \mu_k^{(i)}, \Sigma_k^{(i)})}{\sum_{l=1}^{K} \pi_l^{(i)} \mathcal{N}(\xi_n; \mu_l^{(i)}, \Sigma_l^{(i)})}$$

However, EM algorithm is proven to converge to local optima rather than global optima, in other words, selection of initial parameter $\theta^{(0)}$ seriously affects the estimated result. To tackle this issue, as line 2-3 in algorithm 1 indicate, k-Means clustering process is carried out in advance to initialize the parameters to be learned by EM algorithm.

To retrieve a smooth robot assembly motion trajectory from the learned GMM, Gaussian mixture regression (GMR) algorithm is carried out by taking the decaying term $x : 0 \to 1$ as the *query variable*. Recall that each entry ξ_n is regarded as a concatenated vector $\xi_n = [x_n, y_n^T]^T$ and the regression is predicting $\bar{y} = E(y|x)$. With the conditioning property of multivariate Gaussian distribution in mind, it can be computed as:

$$\bar{y} = \sum_{k=1}^{K} h_k(x)(\mu_{k,y} + \Sigma_{k,yx}\Sigma_{k,x}^{-1}(x - \mu_{k,x})) \tag{4}$$

$$\text{var}(y) = \sum_{k=1}^{K} h_k(x)^2 \left(\Sigma_{k,y} - \Sigma_{k,yx}\Sigma_{k,x}^{-1}\Sigma_{k,yx}^T\right) \tag{5}$$

where

$$h_k(x) = \frac{\pi_k \mathcal{N}(x; \mu_{k,x}, \Sigma_{k,x})}{\sum_{l=1}^{K} \pi_l \mathcal{N}(x; \mu_{l,x}, \Sigma_{l,x})}$$

C. Policy Reinforcement by Local Human Correction

Under precision requirement, robot assembly motion trajectory generated by GMR always fails to work well. This work propose a policy reinforcement strategy that taking local human correction $\tilde{\xi}$ as input to adjust the assembly policy modeled by GMM. As stated by line 9-11 in algorithm 1, when an unexpected collision occurred during robot assembly process the robot aborts the task and keeps its position where the collision occurred. At the same time the corresponded decaying step $\tilde{x}$ is recorded (line 9). Afterwards, the human operator is requested to perform the correction demonstration $\tilde{\xi}$ from the collision position. Here what the operator should do is just move the robot to the *right* position at decaying step $\tilde{x}$. Compared with requiring operators providing extra demonstrations [13], providing corrections locally is more user-friendly. Moreover, as the unexpected collision always occurs in particular poses where relative high precision is required, local corrections turn out to be effective demonstrations of critical poses. Given the collision decaying step $\tilde{x}^{(i)}$ and correction demonstration $\tilde{y}^{(i)}$, the GMM learned in the unsupervised policy learning phase is adjusted by adding another Gaussian distribution with parameters $\theta_{(K+i)} = \{\pi_{K+i}, \mu_{K+i}, \Sigma_{K+i}\}$ (line 11):

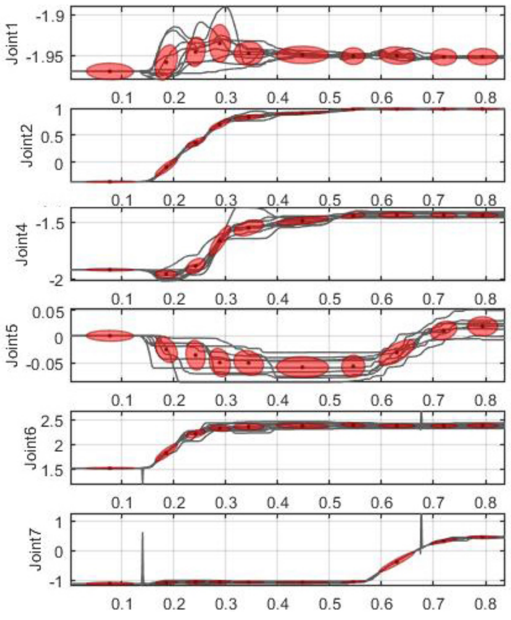
$$\pi_{K+i} = \frac{\min_l(\pi_l)\alpha}{\sum_{l=1}^{K}\pi_l + \min_l(\pi_l)\alpha} \tag{6}$$

$$\mu_{K+i} = [\tilde{x}, \tilde{y}^T]^T \tag{7}$$

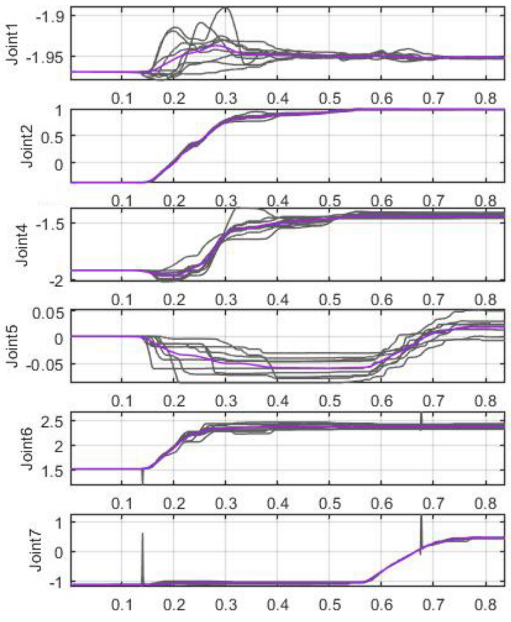
$$\Sigma_{K+i} = \begin{bmatrix} \sigma_{K+i,x} & \mathbf{0} \\ \mathbf{0} & \Sigma_{K+i,y} \end{bmatrix} \tag{8}$$

In view of the fact that the prior π_{K+i} in formula (6) represents the probability of point ξ_n is generated by the $(K+i)$-th Gaussian distribution, a local correction should be activated only when trajectory points around the center μ_{K+i} are estimated. Hence, prior of the added Gaussian distribution is set to not larger than the minimal one of the original GMM by setting $\alpha \in (0,1]$ avoiding greatly impacting the regression. Anyway, the constraint $\sum_{k=1}^{K+i}\pi_k = 1$ must be satisfied at meantime. Center of the added Gaussian distribution is exactly set to be the correction decaying step $\tilde{x}$ concatenated with the ultimate position of the correction as shown in formula (7). $\Sigma_{K+i,y} \in \mathbb{R}^{(D-1)\times(D-1)}$ in formula (8), which stands for the variance of position variable of the added Gaussian distribution and can be viewed as a sign of position variance allowed for the demonstrated task [17], is set to be a diagonal matrix since the local correction can only supply information of an exact position but not the correlation among each of dimensions of the position variable. Furthermore, $\sigma_{K+i,x} > 0$ in formula (8) indicates the variance of the decaying (time) variable of the added distribution. It plays a critical role when the assembly trajectory is retrieved by GMR as we take the decay term x as query variable. Heuristically, its value should be set large enough to activate the added distribution when the regression process is carried out around the correction decaying step $\tilde{x}$. However, too large $\sigma_{K+i,x}$ will impact the regression process too early and too late. In this work, a tradeoff is made by setting

$$\sigma_{K+i,x} = \frac{1}{3}\min(|\tilde{x} - \tilde{x}^-|, |\tilde{x} - \tilde{x}^+|)$$



(a)



(b)

Fig. 4. The learned GMM (a) and the trajectory generated by GMR (b) in joint space

where $\tilde{x}^-$ and $\tilde{x}^+$ are the maximal decaying variable less than $\tilde{x}$ and the minimal one larger than $\tilde{x}$ in the centers of the original GMM, respectively. By doing so, another smooth assembly trajectory is retrieved from the adjusted GMM. Effectiveness of the policy reinforcement strategy can be checked in section III-B.

III. Experiment

A. Experiment Settings

In the experiment, the Franka Panda robot shown in Fig. 1 is used for kinesthetic teaching. Kinesthetic teaching is simple technique to acquire assembly demonstration data as

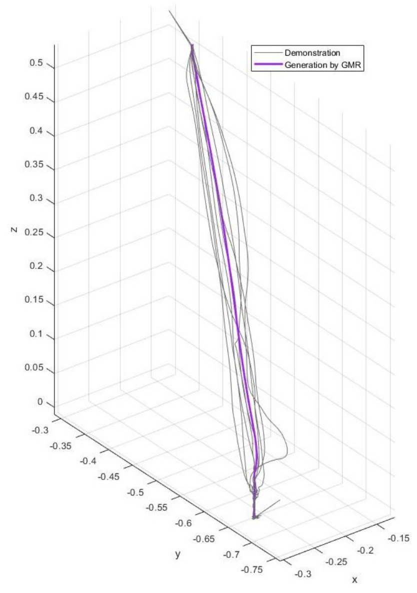
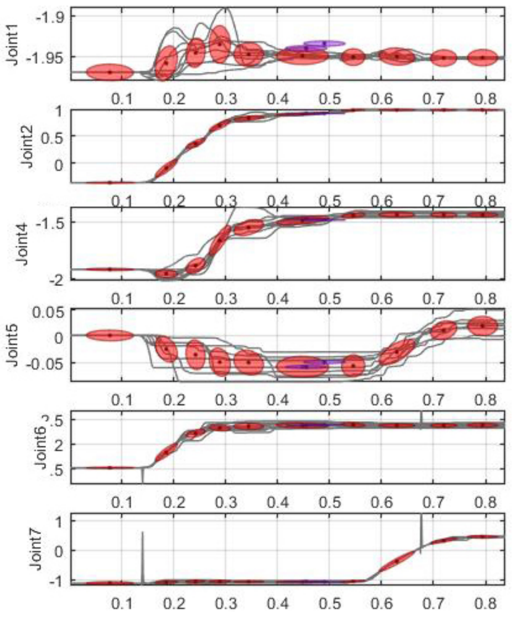


Fig. 5. Demonstrations and the assembly trajectory generated by GMR in Cartesian space

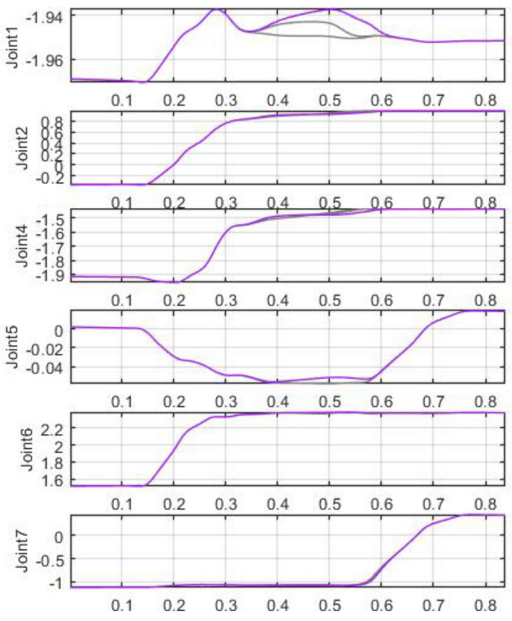
the joint positions can be recorded directly by the encoders in each joint of the robot, that is to say, there is no *correspondence issue* [7] to be considered during the policy learning process. Fig. 7 shows the designed components (Fig. 7(a)) and their assembly relation (Fig. 7(b)) with 2 millimeter clearance in each direction. To accomplish their assembly there must be an insertion motion followed by a rotation motion. It is assumed that one of the components has been picked by the robot. The other one is fixed to a stationary platform. How to grip the components is beyond the scope of this study. The human operator provides multiple demonstrations by kinesthetic teaching. Then the robot follows the assembly trajectory generated by the learned assembly policy. When there comes a collision during the insertion or rotation process, the robot aborts the motion automatically and the operator provides a correction by dragging the end of the robot to the right position. After a few times of corrections, the robot will be able to carry out the assembly task without any trouble. Details of the experiment can also be found in the supplemented video.

B. Experiment Results

Figure 4 shows the learned GMM (the red translucent ellipsoids) in the unsupervised policy learning phase and the generated assembly motion trajectory (the purple curve) by GMR. A smooth motion trajectory is retrieved from the demonstrations. Note that the Franka Panda robot is a 7-Dof robot but we only need 6 Dofs to carry out the illustrated assembly task. In this experiment, the thrid joint of the robot is tried to be fixed during the demonstration process. Fig. 5 shows the generated motion trajectory in Cartesian space. It can be seen that some disturbance introduced by human's trembling or hesitation does not impact the generated trajectory significantly. This is a valuable property



(a)



(b)

Fig. 6. The adjusted GMM (a) and the new trajectory generated by GMR (b) in joint space

of the GMM + GMR framework as humans carry out a precise assembly task with a visual servo method. It is a closed-loop control method which can bring in overshot. However, when the same task is carried out by a robot, it is unnecessary for the robot to follow the same overshot.

In the experiment, the operator performed correction twice to reinforce the GMM to generate feasible robot assembly motion trajectories. By the policy reinforcement strategy, two Gaussian distributions are added to the assembly policy as shown in Fig. 6(a). The purple curve in Fig. 6(b) exhibits the assembly motion trajectory generated by the reinforced GMM while the grey ones exhibit the trajectory generated

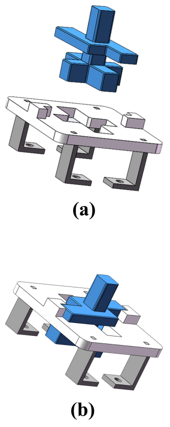


Fig. 7. The components to be assembled in the experiment

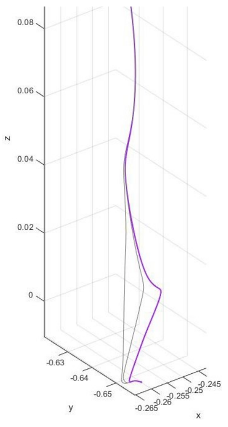


Fig. 8. Robot assembly trajectories generated by the adjusted GMM

from the original GMM and after one correction, respectively. Fig. 8 exhibits the correction as well as the finally generated trajectory in Cartesian space.

As the experiment shows, robot assembly motion trajectory directly generated by GMR algorithm based on the learned GMM does not draw the robot to the target position. After provided with 2 corrections by the operator, the reinforced policy finally generates a suitable robot assembly trajectory in joint space. Note that in this experiment the *target* lies in the same location as the demonstrated one. The proposed method can also derive a policy that generalizes to any other targets in the robot's dexterous workspace through transferring the coordinate to relative coordinate and extend the GMM to task-parametric Gaussian mixture model [15].

IV. Conclusions

For the sake of boosting the success rate of precise assembly perform by robots that are programmed by human demonstration, this paper proposed a Programming by Demonstration with local human correction method. Programming by Demonstration has been regarded as a feasible solution to transfer human skills to the robot. When it comes to assembly tasks with the precision requirement, current methods suffer from a low success rate, which is tackled in this work by a human-in-the-loop framework. The effectiveness of the method is demonstrated by an assembly experiment. As discussed in section II, the variances can be deemed as a sign of precision requirements for each dimension. Although the robot motion trajectory executed eventually is generated by GMR algorithm, in practice, it is in no account necessary to do this since a more efficient motion trajectory can be generated by taking the centers, variances of the GMM and the robot's dynamic properties into account. How an algorithm can generate a smooth motion trajectory that is different from but more efficient than the one by GMR is another future research issue.

References

[1] J. Lambrecht, M. Kleinsorge, M. Rosenstrauch, and J. Krüger, "Spatial programming for industrial robots through task demonstration," *International Journal of Advanced Robotic Systems*, vol. 10, no. 5, pp. 254–264, 2013.
[2] L. Peternel, T. Petrič, and J. Babič, "Robotic assembly solution by human-in-the-loop teaching method based on real-time stiffness modulation," *Autonomous Robots*, vol. 42, no. 1, pp. 1–17, 2018.
[3] F. Suárez-Ruiz and Q.-C. Pham, "A framework for fine robotic assembly," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 421–426.
[4] J. S. Laursen, U. P. Schultz, and L.-P. Ellekilde, "Automatic error recovery in robot assembly operations using reverse execution," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 1785–1792.
[5] J. S. Laursen, L.-P. Ellekilde, and U. P. Schultz, "Modelling reversible execution of robotic assembly," *Robotica*, vol. 36, no. 5, pp. 625–654, 2018.
[6] N. Krüger, A. Ude, H. G. Petersen, B. Nemec, L.-P. Ellekilde, T. R. Savarimuthu, J. A. Rytz, K. Fischer, A. G. Buch, D. Kraft, *et al.*, "Technologies for the fast set-up of automated assembly processes," *KI-Künstliche Intelligenz*, vol. 28, no. 4, pp. 305–313, 2014.
[7] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.
[8] Z. Zhu and H. Hu, "Robot learning from demonstration in robotic assembly: A survey," *Robotics*, vol. 7, no. 2, p. 17, 2018.
[9] C. Celemin and J. Ruiz-del Solar, "Coach: learning continuous actions from corrective advice communicated by humans," in *2015 International Conference on Advanced Robotics (ICAR)*. IEEE, 2015, pp. 581–586.
[10] Y. Schroecker, H. Ben Amor, and A. Thomaz, "Directing policy search with interactively taught via-points," in *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2016, pp. 1052–1059.
[11] C.-Y. Chiu, S.-P. Chao, M.-Y. Wu, S.-N. Yang, and H.-C. Lin, "Content-based retrieval for human motion data," *Journal of Visual Communication and Image Representation*, vol. 15, no. 3, pp. 446–466, 2004.
[12] D. Kulić, W. Takano, and Y. Nakamura, "Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden markov chains," *The International Journal of Robotics Research*, vol. 27, no. 7, pp. 761–784, 2008.
[13] S. Calinon and A. Billard, "Incremental learning of gestures by imitation in a humanoid robot," in *Proceedings of the ACM/IEEE international conference on Human-robot interaction*. ACM, 2007, pp. 255–262.
[14] ——, "A probabilistic programming by demonstration framework handling constraints in joint space and task space," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 367–372.
[15] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," *Intelligent Service Robotics*, vol. 9, no. 1, pp. 1–29, 2016.
[16] M. Kyrarini, M. A. Haseeb, D. Ristić-Durrant, and A. Gräser, "Robot learning of industrial assembly task via human demonstrations," *Autonomous Robots*, vol. 43, no. 1, pp. 239–257, 2019.
[17] D. Berio, F. F. Leymarie, and S. Calinon, "Interactive generation of calligraphic trajectories from gaussian mixtures," in *Mixture Models and Applications*. Springer, 2020, pp. 23–38.
[18] S. Calinon, "Mixture models for the analysis, edition, and synthesis of continuous time series," in *Mixture Models and Applications*. Springer, 2020, pp. 39–57.
[19] H. Akaike, "A new look at the statistical model identification," *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, 1974.
[20] G. Schwarz *et al.*, "Estimating the dimension of a model," *The annals of statistics*, vol. 6, no. 2, pp. 461–464, 1978.
[21] D. J. Spiegelhalter, N. G. Best, B. P. Carlin, and A. Van Der Linde, "Bayesian measures of model complexity and fit," *Journal of the royal statistical society: Series b (statistical methodology)*, vol. 64, no. 4, pp. 583–639, 2002.