The International Journal of Advanced Manufacturing Technology (2022) 119:6775–6787
https://doi.org/10.1007/s00170-022-08652-z

ORIGINAL ARTICLE

A robot learning from demonstration framework for skillful small parts assembly

Haopeng Hu[1] · Xiansheng Yang[1] · Yunjiang Lou[1]

Received: 2 March 2021 / Accepted: 2 January 2022 / Published online: 22 January 2022

Abstract
Increasing demand for higher production flexibility and smaller production batch size pushes the development of manufacturing industry towards robotic solutions with fast setup and reprogram capability. Aiming to facilitate assembly lines with robots, the learning from demonstration (LfD) paradigm has attracted attention. A robot LfD framework designed for skillful small parts assembly applications is developed, which takes position, orientation and wrench demonstration data into consideration. In view of constraints in industrial small parts assembly applications, two cascaded assembly polices are learned from separated assembly demonstration data to avoid potential under-fitting problem. With the proposed assembly policies, reference orientation and wrench trajectories are generated as well as coupled with the position data. Effectiveness of the proposed LfD framework is validated by a printed circuit board assembly experiment with a torque-controlled robot.

Keywords Flexible manufacturing · Learning from demonstration · Robotic assembly

1 Introduction

For the sake of adapting to the ongoing shift from mass production to mass customization of products, manufacturing enterprises have to keep evolving their assembly lines to handle more product variation, shorter product life cycles, and smaller batch sizes [1–3], i.e., developing flexible assembly lines. Current industrial robots are notoriously difficult to program, leading to high change-over time and expert labor consumption. To make robotic assembly applicable to low-volume high-mix production, one approach is to make the programming of assembly tasks so intuitive that can be accomplished by workers without traditional robot programming skills [4, 5]. Learning from demonstration (LfD) [6], or programming by demonstration, is a paradigm that aims to transfer human's skills to robots by human demonstration instead of the unintuitive and tedious robot programming by expert robot users [7]. The strength of LfD pays in industrial assembly applications as some assembly processes are such complicate that can be neither easily scripted nor easily defined as an optimization problem, but can be easily demonstrated [8]. With LfD, one or more assembly policies can be learned from the demonstration data which can also generate robots' actions under new conditions, i.e., enable robots' adaptive behavior [9].

The robotic assembly processes, as exhibited in Fig. 1, can be divided into two consecutive phases based on the density feature of the demonstration data and variance of the demonstration trajectories. When the human operator performs an assembly demonstration, he/she may guide the end-effector from anywhere convenient to approach the fixed part as soon as possible. As a result, demonstration data in this process feature low density. Moreover, since the initial pose is random, profiles of the demonstrated trajectories deviate significantly from each other in this process. Afterwards, the human operator carefully adjusts the pose of the part in hand to a specific pose that is convenient for assembling the two parts. Then, the human operator assembles the part by delicately moving the end-effector. As a result, density of the data after the specific pose is relatively high. In this process, there are only small deviations among trajectories because they are constrained by the design of the two parts. Therefore, the two phases exhibit quite different data densities and trajectory variances. By taking the pose

This work was supported in part by NSFC-Shenzhen Robotics Basic Research Center Program Grant U1713202.

✉ Yunjiang Lou
louyj@hit.edu.cn

[1] School of Mechanical Engineering and Automation, Harbin Institute of Technology Shenzhen, Shenzhen, China

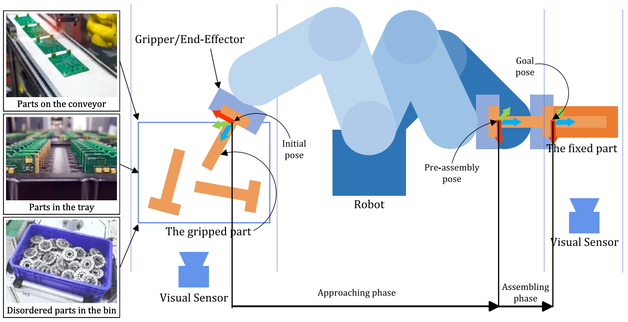


Fig. 1 The whole assembly process consisting of the approaching and the assembling phases

of the end-effector where the gripped part starts contacting with the fixed part as the cutting point, the assembly process before it is defined as the *approaching phase* and that after it is defined as the *assembling phase*. The cutting point is called the *pre-assembly pose*, as shown in Fig. 1. During the approaching phase, end-effector of the robot moves to the pre-assembly pose after picking up one part from some vessel, and the robot assembles one part to the other that is fixed somewhere in the assembling phase.

In the literature, one formulation of LfD-based parts assembly tasks is the *peg-in-hole* problem [10–12] which regards the assembly process as inserting a peg into a hole and the clearance between them is tight. In those works, the pegs (the gripped parts in Fig. 1) are always assumed to have been placed in contact with the holes (the fixed parts in Fig. 1). Those works focus on compensation of the motion error and relate to the assembly process in the assembling phase. Reinforcement Learning-based methods are widely applied to learn a decision making policy that maps states to actions through trial-and-error [13–16]. However, assembly motion during the assembling phase of skillful assembly tasks [8] may not be just inserting movement. Some LfD studies upon assembly applications regard the assembly tasks as the robot taking one part from somewhere and moving to some given goal pose [10, 17, 18]. A pre-structured *policy* [6] is learned through supervised [10] or unsupervised [17, 18] machine learning algorithm. Given novel initial and goal pose of the part, the policy generates smooth robot assembly motion trajectories or real-time control input to drive the robot to the goal pose. This formulation reduces the assembly tasks to a *pick-and-place* problem and can be deemed as a process in the approaching phase. To the best knowledge of the authors, no studies tried to provide an overall solution for the WHOLE assembly process, consisting of the consecutively approaching and assembling phases. The fundamental difficulty lies in how to formulate the two distinct phases by a single unified framework.

In this work, we focus on the small part assembly tasks and regard the robot LfD problem as a policy learning problem given multiple demonstrated robot assembly motion trajectories of one object. To accomplish such assembly tasks with robots, two constraints must be taken into account, namely the *environment constraint* and the *object constraint*. In the industrial fields, the environment constraint is introduced by the deployment of industrial agents such as placement of robots, sensors, and conveyors. It varies from line to line, or even from day to day in re-configurable assembly lines [2] and allows of large variation in the assembly motion of robots. Hence, the generalization ability of the learned policy matters in the approaching phase. The object constraint includes the assembly motion (position and orientation trajectories) required and the corresponding force and torque pattern throughout the assembly process. It is totally defined by the design of product. Therefore, generalization ability of the learned policy in the assembling phase is not so important as that in the approaching phase. By improving the task-parameterized Gaussian mixture model (TP-GMM) and task-parameterized Gaussian mixture regression (TP-GMR) algorithm, a unified robot LfD framework is tailored for skillful small parts assembly in this work. Two cascaded policies in Cartesian space are learned from separated demonstration data for the approaching and assembling phases, respectively. Contributions of this work go as follows. By taking the two constraints into account, a novel unified LfD framework for assembly applications that governs an assembly task as two assembly phases is developed with two cascaded policies that coupled position, orientation and force-torque (wrench) are learned separately to avoid the under-fitting problem. A switch strategy between the two policies is also applied to avoid unexpected jump between the trajectories generated by the two polices. The assembly policies are designed to couple the orientation and wrench information with the generated position trajectories which enjoys more robustness over the unique query variable trick and enables online trajectory adaptation.

Remained content of this article is organized follows. Section 2 expounds the proposed robot LfD assembly framework from system overview (Section 2.1) to policy learning (Section 2.2). Experiment verification of it lies in Section 3. Section 4 concludes this article.

2 Methodology

2.1 Framework overview

Work flow of the proposed robot LfD assembly framework is exhibited in Fig. 2. A typical LfD framework always contains three work steps, namely demonstration, policy learning and policy execution (under strange environments), that is, the first decision to be made when designing a LfD system is which demonstration technique is used given specific

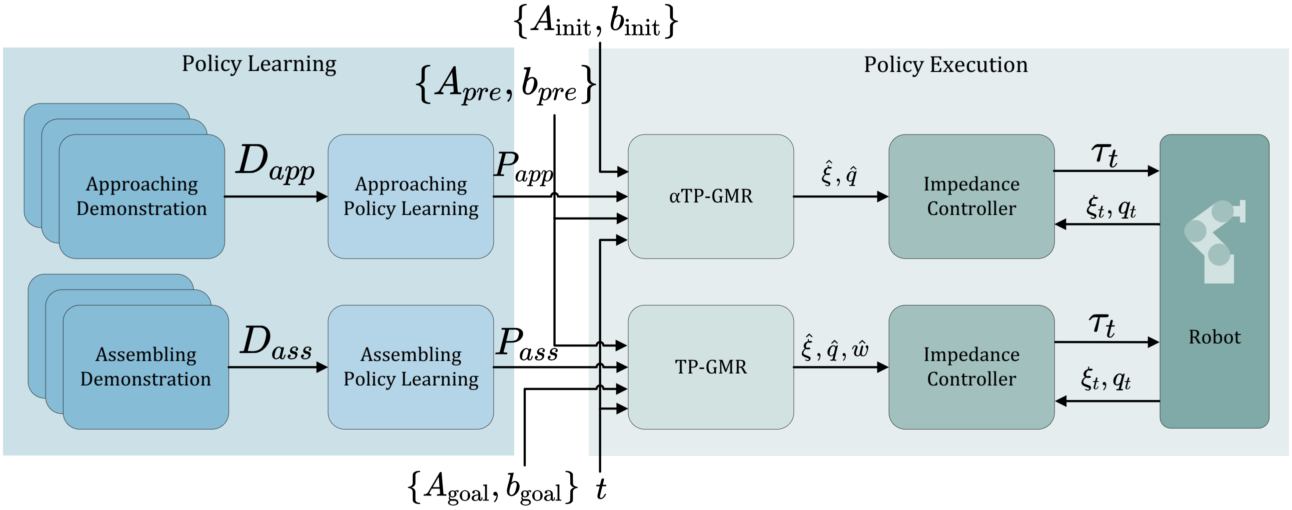


Fig. 2 Work flow of the robot LfD assembly framework

applications [6]. Here the *kinesthetic teaching* technique is selected for the sake of its usability for non-expert robot users. During demonstration, the operator successively performs the approaching motion and assembling motion. Afterwards, two assembly policies $P_{app/ass}$ are learned given the separated demonstration data $D_{app/ass}$. During the policy execution process, the learned polices serve to generate both position $\hat{\xi}$ and orientation $\hat{q}$ motion trajectories and the force/torque profile given new *task parameters* $\{A, b\}$. As for robots' execution of the policies given the initial, pre-assembly and goal pose, a Cartesian impedance controller, which takes the joint torque τ as control input, serves to enact those policies.

2.2 Assembly policy

By an assembly policy, what we mean here is an assembly skill model that takes the initial pose, goal pose as inputs while generates a suitable robot motion trajectory as output. In assembly tasks, both position and orientation play important roles while the force and torque information (wrench) also matters. Policies related to position, orientation and wrench are separately learned but coupled in this work.

2.2.1 Position policy

Three elements ought to be considered to pre-structure the policy for assembly skill learning. First, as for modeling the assembly skills, the *location primitives* [18] are preferable to the *trajectory primitives*. This is because the robot ought to try to imitate the position/orientation pattern of the assembly demonstration data rather than the velocity pattern. Second, the *stochastic models* [6] instead of *deterministic models* are essential because there are always variances among each demonstration trajectories. And last but not least, the *generalization capability* [19] of the model is necessary since the task parameters may vary during both the policy learning and policy execution process. On accounting of these elements, the *task-parameterized Gaussian mixture model* (TP-GMM) [20] is taken to pre-structure the assembly policies and task-parameterized Gaussian mixture regression (TP-GMR) algorithm is used to generate the motion trajectories. Unfortunately, given the assembly tasks in Fig. 1 the classical TP-GMM together with TP-GMR algorithm exhibits poor performance owing to the two distinct phases. As shown in Fig. 8 GMM-based policies encapsulate the demonstration trajectories with multiple Gaussian distributions. Once there are small number of Gaussian distributions ($K = 8$ in the comparison experiment), the policy may fail to generate applicable trajectories. This is because there are not enough distributions to encapsulate the delicate motion in the assembling phase. In this case the policy is unreliable, let alone capable of generalization. However, the larger the number of Gaussian distribution is, the more redundant motion will be generated. In this case, the policy is inefficient and will lead to extra wearing of the robot. In summary, it is difficult to balance the efficiency and reliability of the classical TP-GMM-based policy.

Hence, in this work two parametric policies pre-structured by TP-GMM are learned respectively to model the two phases of the demonstrated assembly skill and generate the assembly motion trajectory by *α*TP-GMR [19]. With TP-GMM, the position trajectory distribution $P(\xi)$ of the assembly skill in the j-th *task frame* is modeled by

$$P\left(\xi^{(j)}\right) = \sum_{k=1}^{K} \pi_k \mathcal{N}\left(\xi^{(j)} \middle| \mu_{\xi,k}^{(j)}, \Sigma_{\xi,k}^{(j)}\right)$$

where $\xi^{(j)} \triangleq [t, x^{(j)}, y^{(j)}, z^{(j)}]^\top$ is the position in the j-th task frame at time/phase t, π_k is the *prior* of the k-th multivariate Gaussian distribution with $\sum_{k=1}^{K} \pi_k = 1$, $\mu_{\xi,k}^{(j)} \in \mathbb{R}^4$ and $\Sigma_{\xi,k}^{(j)} \in \mathbb{R}^{4\times4}$ as centers and co-variances. The parameters of TP-GMM are summarized as

$$\left\{ \pi_k, \left\{ \mu_{\xi,k}^{(j)}, \Sigma_{\xi,k}^{(j)} \right\}_{j=1}^{J} \right\}_{k=1}^{K} \tag{1}$$

The so-called *task frames*, or *task parameters*, are given by the J task frames $\{A_j, b_j\}_{j=1}^{J}$ that are time-invariant in this work and practical industrial assembly tasks,

$$A_j = \begin{pmatrix} 1 & \mathbf{0} \\ \mathbf{0} & R_j \end{pmatrix}, \quad b_j = \begin{pmatrix} 0 \\ p_j \end{pmatrix} \tag{2}$$

in which $R_j \in SO(3)$ and $p_j \in \mathbb{R}^3$ are rotation matrix and translation vector of frame j with respect to the world frame. Given the demonstration data $D : \{\xi_n\}_{n=1}^{N}$ in the world frame, rather than directly used to estimate the parameters, it is transformed to J task frames firstly through

$$\xi_n^{(j)} = A_j^{-1}(\xi_n - b_j)$$

Parameters in (1) are estimated in each frame individually by Expectation-Maximization algorithm in which the joint distribution of data in J task frames are used to compute the expectation (E-Step) but parameters are estimated only with the data in the corresponding frame (M-Step).

By taking time/phase variable t as *query variable*, given the task frames (2), a retrieved or generalized assembly motion trajectory will be computed through TP-GMR algorithm [20]. Anyway, generalization (interpolation and extrapolation) capability of the approaching policy counts for much more than that of the precise assembly policy. In order to further boost the generalization capability of the approaching policy P_{app}, i.e. to preserve the *local structure* of demonstration data around the approaching frame, the frame-weighted TP-GMR algorithm (αTP-GMR) [19] is utilized here instead of the typical one. In the j-th frame, the Gaussian center μ and covariance Σ can be partitioned as

$$\mu_{\xi,k}^{(j)} = \begin{pmatrix} \mu_{t,k}^{(j)} \\ \mu_{\xi_{\mathcal{O},k}}^{(j)} \end{pmatrix}, \quad \Sigma_{\xi,k}^{(j)} = \begin{pmatrix} \Sigma_{t,k}^{(j)} & \Sigma_{t\xi_{\mathcal{O}},k}^{(j)} \\ \Sigma_{\xi_{\mathcal{O}}t,k}^{(j)} & \Sigma_{\xi_{\mathcal{O},k}}^{(j)} \end{pmatrix}$$

and the trajectory distribution in the jth frame generated by GMR is given by

$$P\left(\hat{\xi}_{\mathcal{O},n}^{(j)} | t_n\right) = \mathcal{N}\left(\sum_{k=1}^{K} \phi_k(t_n) \hat{\xi}_{\mathcal{O},k,n}^{(j)}, \sum_{k=1}^{K} \phi_k^2(t_n) \hat{\Sigma}_{\xi_{\mathcal{O}},k,n}^{(j)} \right) \tag{3}$$

where

$$\hat{\xi}_{\mathcal{O},k,n}^{(j)} = \mu_{\xi_{\mathcal{O}},k}^{(j)} + \Sigma_{\xi_{\mathcal{O}}t,k}^{(j)} \left(\Sigma_{t,k}^{(j)}\right)^{-1} \left(t_n - \mu_{t,k}^{(j)}\right)$$

$$\hat{\Sigma}_{\xi_{\mathcal{O}},k,n}^{(j)} = \Sigma_{\xi_{\mathcal{O}},k}^{(j)} - \Sigma_{\xi_{\mathcal{O}}t,k}^{(j)} \left(\Sigma_{t,k}^{(j)}\right)^{-1} \Sigma_{t\xi_{\mathcal{O}},k}^{(j)}$$

$$\phi_k(t_n) = \frac{\pi_k \mathcal{N}\left(t_n | \mu_{t,k}, \Sigma_{t,k}\right)}{\sum_{l=1}^{K} \pi_k \mathcal{N}\left(t_n | \mu_{t,l}, \Sigma_{t,l}\right)}$$

Then the generated trajectory distribution in the world frame is estimated by

$$\mathcal{N}\left(\hat{\xi}_{\mathcal{O},n}, \hat{\Sigma}_{\xi_{\mathcal{O}},n}\right) \propto \prod_{j=1}^{J} \mathcal{N}\left(R_j \hat{\xi}_{\mathcal{O},n}^{(j)} + p_j, \frac{1}{\alpha_n^{(j)}} R_j \hat{\Sigma}_{\xi_{\mathcal{O}},n}^{(j)} R_j^T \right)$$

in which $\hat{\xi}_{\mathcal{O}}$ and $\hat{\xi}_{\mathcal{O}}^{(j)}$ are *queried variable* by GMR in world frame and the j-th task frame respectively ($[x, y, z]^T$ in this work), and $\hat{\Sigma}_{\xi_{\mathcal{O}}}$ stands for the corresponding co-variance. $\alpha_n^{(j)}$ indicates the *weight* of the j-th task frame at step n,

$$\alpha_n^{(j)} = \frac{\left\| \left(\tilde{\Sigma}_n^{(j)}\right)^{-1} \right\|}{\sum_{l=1}^{J} \left\| \left(\tilde{\Sigma}_n^{(l)}\right)^{-1} \right\|} \tag{4}$$

$\tilde{\Sigma}_n^{(j)}$ in (4) is the covariance of demonstration data in the j-th task frame at time step n, computation of which costs additional effort but only required once after demonstration. As for the assembling policy P_{ass}, typical TP-GMR algorithm ($\alpha_n^{(j)} = 1$) will work well since there is little variation in the profile of position trajectories in the assembling phase.

Anyway, generating the trajectories with two polices does not means simply performing the TP-GMR algorithm one by one. As indicated in Fig. 2, the pre-assembly frame $\{A_{pre}, b_{pre}\}$ is given in advance but destination of the motion generated by the approaching policy will not exactly be the pre-assembly pose. There may exist unexpected *gap* between the motion trajectories between the two phases which leads to redundant motion. To deal with this issue, a switch strategy between the two policies is proposed. To begin with, the last pose of the trajectories generated by the approaching policy is taken as the *dummy pre-assembly frame* utilized for the assembling policy. After generation of the trajectory of the whole assembly process, we calculate two cutting points by

$$\hat{\xi}_{\mathcal{O},\tilde{n}_{app}} = P_{app}\left(\arg\max_k \mu_{t,k}^{(j)}\right)$$

$$\hat{\xi}_{\mathcal{O},\tilde{n}_{ass}} = P_{ass}\left(\arg\min_k \mu_{t,k}^{(j)}\right)$$

where $\mu_{t,k}^{(j)}$ refers to the first entry of the center $\mu_k^{(j)}$ of the TP-GMM (1). That is to say, the center $\mu_k^{(j)}$ of the temporally last

Gaussian distribution of the approaching policy and that of the temporally first Gaussian distribution of the assembling policy are taken as the cutting points. Note that the first entry of the Gaussian center $\mu_{\xi,k}^{(j)}$ is the time/phase variable that are identical in each task frame. Then we interpolate $\tilde{n}_{ass}$ positions between the two cutting points while get rid of the trajectories temporally after $\hat{\xi}_{\mathcal{O},\tilde{n}_{app}}$ and those before $\hat{\xi}_{\mathcal{O},\tilde{n}_{ass}}$.

2.2.2 Orientation policy

As one of the key aspects, it is inevitable to take the orientation trajectories required in many assembly tasks. In the literature, unit quaternion $q \triangleq [q_w, q_x, q_y, q_z]^\top$ is a popular representation of orientation in LfD studies [10, 18, 21], which is a kind of non-minimal representation defined on an unit sphere manifold $\mathcal{S}^3$ whose tangent space $\mathcal{T}_q\mathcal{S}^3$ locally linearizes the manifold and behaves like Euclidean space $\mathbb{R}^3$ [22]. Compared with minimal representations of orientation such as Euler angle, it benefits from no singularity. By taking the orientation of the pre-assembly or goal pose as *auxiliary quaternion* q_a [21], the demonstrated orientation data q can be transformed to the tangent space of q_a through the *logarithmic map* $\mathcal{S}^3 \to \mathcal{T}_{q_a}\mathcal{S}^3$

$$\eta \triangleq \log(\tilde{q}) = \begin{cases} \arccos(\tilde{q}_w)\dfrac{\tilde{\mathbf{u}}}{\|\tilde{\mathbf{u}}\|}, & \|\tilde{\mathbf{u}}\| \neq \mathbf{0} \\ [0,0,0]^\top, & \text{otherwise} \end{cases} \tag{5}$$

where $\tilde{q} = q * \bar{q}_a \triangleq \left[\tilde{q}_w, \tilde{\mathbf{u}}^\top\right]^\top$ with $\tilde{\mathbf{u}} \triangleq [\tilde{q}_x, \tilde{q}_y, \tilde{q}_z]^\top$ is the vector part of unit quaternion. $\bar{q}_a$ denotes the *quaternion conjugation* of q_a. $*$ represents the *quaternion multiplication* operation [23]. $\eta \in \mathcal{T}_{q_a}\mathcal{S}^3 \subset \mathbb{R}^3$ is the projection of q in the tangent space of q_a.

Afterwards, policy learning (EM algorithm) and trajectory generation (GMR) processes are carried out on $\mathcal{T}_{q_a}\mathcal{S}^3$ with Euclidean distance $dist(\eta_1, \eta_2) = \|\eta_1 - \eta_2\|$. Finally, orientation trajectories generated by the GMR ought to be projected back to the unit sphere manifold S^3 via the *exponential map* $\mathcal{T}_{q_a}\mathcal{S}^3 \to \mathcal{S}^3$

$$\tilde{q} = \exp(\eta) = \begin{cases} \cos(\|\eta\|) + \sin(\|\eta\|)\dfrac{\eta}{\|\eta\|}, & \eta \neq 0 \\ [1,0,0,0]^T, & \text{otherwise} \end{cases} \tag{6}$$

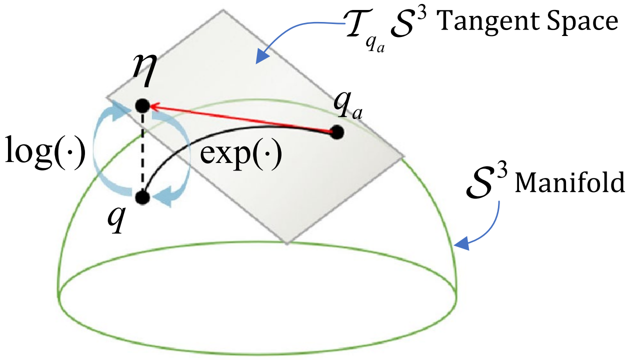


Fig. 3 Graphic representation of the relationship between q and η

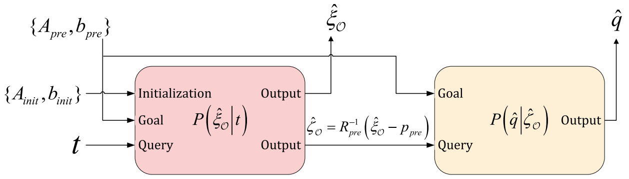

$\{A_{pre}, b_{pre}\}$
$\hat{\xi}_{\mathcal{O}}$
$\hat{q}$
$\{A_{init}, b_{init}\}$
Initialization
Output
Goal
$P(\hat{\xi}_{\mathcal{O}}|t)$
t
Query
Output
$\hat{\zeta}_{\mathcal{O}} = R_{pre}^{-1}(\hat{\xi}_{\mathcal{O}} - p_{pre})$
Goal
$P(\hat{q}|\hat{\zeta}_{\mathcal{O}})$
Output
Query

Fig. 4 Structure of the approaching policy

Figure 3 exhibits the relationship between q and η. Note that both (5) and (6) are defined with respect to the same auxiliary unit quaternion q_a.

In many assembly tasks, the coupling of position and orientation must be taken into consideration. In $\mathcal{S}^3$ manifold, it is also feasible to make use of TP-GMM for generalization to new goal orientation [24]. By doing so, translation and rotation movements are coupled by the unique query variable. However, it is evident that during the approaching phase, orientation of the end-effector does not matter in the early phase but does matter around the vicinity of pre-assembly pose. Therefore, in this work the positions with respect to the pre-assembly and goal frame act as query for retrieving or generalizing the trajectories of orientation, i.e. rotation and translation movements are coupled by the covariance of multi-variate Gaussian distributions. Compared with coupling by the unique query variable, strength of the proposed coupling strategy lie in the fact that the robot may not exactly follow the generated assembly trajectories owing to the performance of controller, noise of sensors and perturbation, which makes online adaptation of orientation plays a valuable role. By doing so, we estimate the end-effector's orientation given the current position rather than current time/phase. Figures 4 and 5 indicate the proposed coupling technique and describe the structure of the approaching policy and assembling policy respectively. Taking the output

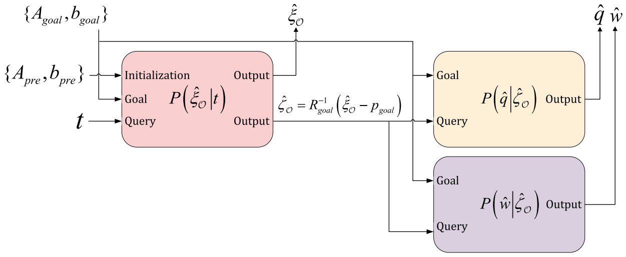

$\{A_{goal}, b_{goal}\}$
$\hat{\xi}_{\mathcal{O}}$
$\hat{q}$ $\hat{w}$
$\{A_{pre}, b_{pre}\}$
Initialization
Output
Goal
$P(\hat{\xi}_{\mathcal{O}}|t)$
t
Query
Output
$\hat{\zeta}_{\mathcal{O}} = R_{goal}^{-1}(\hat{\xi}_{\mathcal{O}} - p_{goal})$
Goal
$P(\hat{q}|\hat{\zeta}_{\mathcal{O}})$
Output
Query
Goal
$P(\hat{w}|\hat{\zeta}_{\mathcal{O}})$
Output
Query

Fig. 5 Structure of the assembling policy

of (3) in pre-assembly frame or goal frame as query variable, state of the GMM is defined as $\varphi \triangleq \left[\xi_{\mathcal{O}}, \eta\right]^T$, i.e. parameters of the GMM are $\left\{\pi_{\varphi,k}, \mu_{\varphi,k}, \Sigma_{\varphi,k}\right\}_{k=1}^{K}$. Similar to (3), the orientation trajectory distribution generated by GMR turns out to be

$$P\left(\hat{\eta}_n \middle| \hat{\xi}_{\mathcal{O},n}\right) = \mathcal{N}\left(\sum_{k=1}^{K} \phi_k\left(\hat{\xi}_{\mathcal{O},n}\right)\hat{\eta}_{k,n}, \sum_{k=1}^{K} \phi_k\left(\hat{\xi}_{\mathcal{O},n}\right)^2 \hat{\Sigma}_{\eta,k,n}\right) \quad (7)$$
$$\hat{q}_n = \exp(\hat{\eta}_n) * q_a$$

where $\hat{\eta}_n$, $\hat{\Sigma}_{\eta,k,n}$, $\phi_k(\cdot)$ are counterparts of those in (3) and the number of Gaussian distributions K may not be the same as that in (1).

2.2.3 Wrench policy in the assembling phase

A typical assembly operation requires the knowledge of not only position and orientation trajectories but also the accompanying wrench profiles for successful assembly [10]. There is no probabilistic model in terms of the wrench w in the approaching policy since the end-effector of the robot is moving through the air as indicated by Fig. 1. In the assembling phase, the wrench pattern is modeled by GMM. As shown in Fig. 5, the wrench policy also takes position with respect to the goal frame as query variable. Wrench trajectory distribution generated by GMR shares the similar policy structure and coupling strategy as (7) just by replacing $\hat{\eta}$ by $\hat{w}$

$$P\left(\hat{w}_n \middle| \hat{\xi}_{\mathcal{O},n}\right) = \mathcal{N}\left(\sum_{k=1}^{K} \phi_k\left(\hat{\xi}_{\mathcal{O},n}\right)\hat{w}_{k,n}, \sum_{k=1}^{K} \phi_k\left(\hat{\xi}_{\mathcal{O},n}\right)^2 \hat{\Sigma}_{w,k,n}\right) \quad (8)$$

Anyway, there is no need to exactly follow the generated wrench trajectory during assembly processes [10]. In this work, the queried wrench trajectory distribution (8) plays two roles. First and foremost, it provides a reference wrench pattern for successful execution of the task since the parts require enough wrench to be assembled. Secondly, it also serves as an indicator of possible damaging of the parts. Variability of the demonstrations is encapsulated in the covariance matrices of the trajectory distribution which can be exploited to detect if the robot reaches an unexpected pose [25]. By eigenvalue decomposition of covariance matrix $\hat{\Sigma}_{w,n} = V\Lambda_{w,n}V^{-1}$, each eigenvalue $\lambda_{i,t}$ corresponds to the allowable variability of the force/torque term at time/phase t_n. In the experiment, we stop motion of the robot once any one dimension of the current external wrench $w_{i,t}$ exceeds the limit

$$\left[\hat{w}_{i,t} - \beta|\lambda_{i,t}|, \hat{w}_{i,t} + \beta|\lambda_{i,t}|\right], \beta > 1$$

3 Experiment verification

3.1 Experiment setup

In this section, a printed circuit board (PCB) [26] assembly task is selected to demonstrate the work flow of the proposed robot LfD assembly framework and validate its effectiveness. As shown in Fig. 6, to successfully assemble the PCB to its bottom case, after plugging it through the scroll wheel holder the robot ought to carry out an insertion motion and then fit the PCB to the locating pins while pressing it to make it fastened by the resilient fasteners. The tolerance between the PCB and the bottom case is 0.5 ± 0.1mm in the assembly process. A torque-controlled Franka Emika 7-DoF robot, whose position repeatability is ± 0.1mm, serves as the policy executor. A structured light camera works to acquire the goal frame $\{A_{goal}, b_{goal}\}$ in the experiment, as exhibited by Fig. 7. The structured light camera is capable of obtaining 6-D pose of the bottom case. The initial pose of the PCB is obtained by forward kinematics of the robot. Figure 7 also demonstrates the kinesthetic teaching process. To begin with, the operator guides the robot with gravity compensation by the *pilot* on the robot's 7th joint from arbitrary initial pose to the pre-assembly pose (the approaching phase). Afterwards, through carefully adjusting the pose of the end-effector to guarantee safety, the operator performs the PCB assembly task (the assembling phase).

In both the approaching and the assembling phases, the number of task frames $J = 2$ in (1) as indicated by Fig. 2. The goal frame can be estimated via observing the pose of the bottom case with the 3D camera while the pre-assembly frame $\{A_{pre}, b_{pre}\}$ is calculated with the constant transformation from the goal frame to it. Let $\{p_{pre}^{(m)}, q_{pre}^{(m)}\}_{m=1}^{M}$ and $\{p_{goal}^{(m)}, q_{goal}^{(m)}\}_{m=1}^{M}$ represent the M demonstrated pre-assembly and goal poses respectively. To estimate the transformation between the two poses, we take the *average* of both poses first. The average positions $\bar{p}_{pre}$ and $\bar{p}_{goal}$ can be simply calculated by

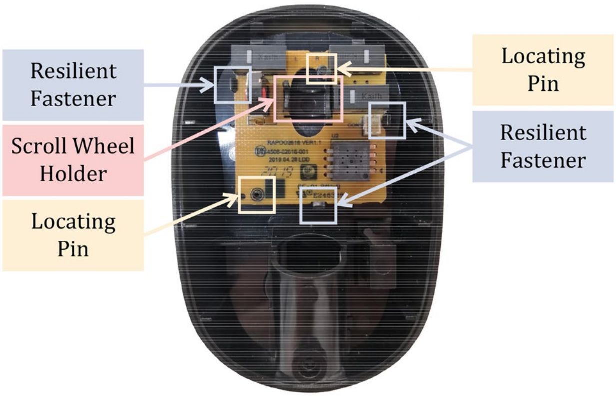


Fig. 6 The PCB and bottom case of a cursor mouse

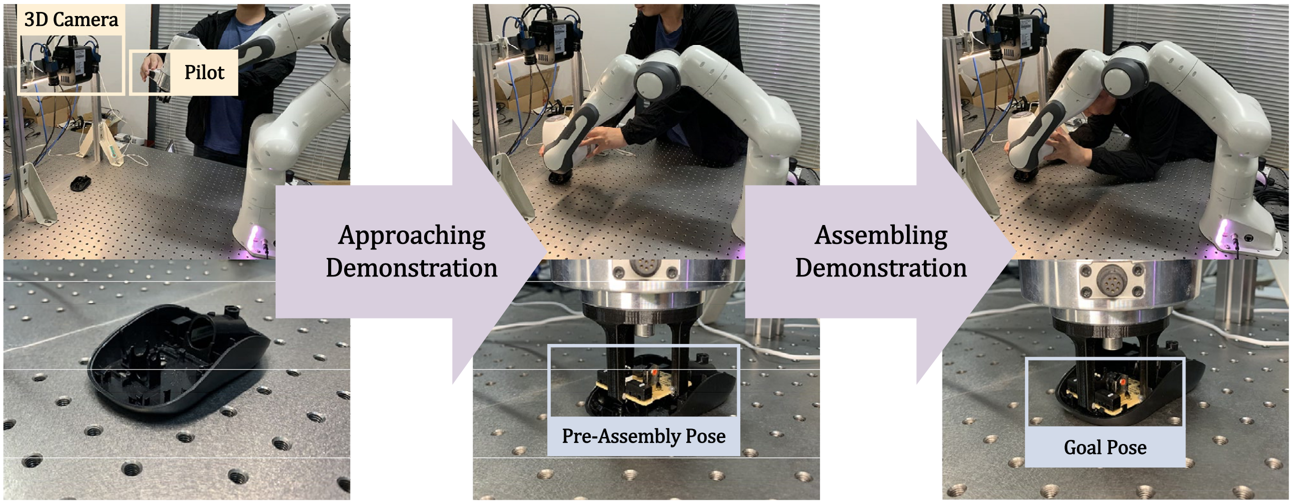


Fig. 7 Kinesthetic guiding demonstration process

$$\bar{p}_{pre} = \frac{1}{M}\sum_{m=1}^{M} p_{pre}^{(m)}, \quad \bar{p}_{goal} = \frac{1}{M}\sum_{m=1}^{M} p_{goal}^{(m)}$$

However, with regards to the average orientations (unit quaternions) we follow the method [27] and calculate

$$Q_{pre} = \sum_{m=1}^{M} q_{pre}^{(m)} \cdot q_{pre}^{(m)\top}, \quad Q_{goal} = \sum_{m=1}^{M} q_{goal}^{(m)} \cdot q_{goal}^{(m)\top} \tag{9}$$

Note that the multiplication operator "·" in (9) represents quaternion dot product [23] (Euclidean inner product for vectors in $\mathbb{R}^4$) rather than quaternion multiplication. Afterwards, by applying principal component analysis (PCA) upon $Q \in \mathbb{R}^{4\times4}$ the normalized eigenvector corresponding to the largest eigenvalue is taken as the average quaternion which can be converted into rotation matrix $\bar{R}$. Then given the estimated goal frame $\{A_{goal}, b_{goal}\}$ and recall the definition (2), we can calculate the pre-assembly frame through

$$A_{pre} = A_{goal}\begin{pmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \bar{R}_{goal}^{\top} \end{pmatrix}\begin{pmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \bar{R}_{pre} \end{pmatrix},$$

$$b_{pre} = A_{goal}\begin{pmatrix} 0 \\ \bar{R}_{goal}^{\top}\bar{p}_{pre} - \bar{R}_{goal}^{\top}\bar{p}_{goal} \end{pmatrix} + b_{goal}$$

Since human operators can never perform one task multiple times within the same period, it is necessary to temporally *align* the demonstration data before policy learning. *Dynamic time wrapping* (DTW) is a populous temporal alignment method in LfD studies [18, 28, 29] including this work. With kinesthetic teaching, the human operator corrupts the wrench profile estimated by joints torques [10] in the assembling phase. To attenuate this effect, in this experiment the robot plays back the demonstrated motion in the assembling phase to obtain clean wrench demonstration data. Moreover, separation of demonstration data is performed manually, nonetheless, automatic separation is also possible by analyzing velocity and acceleration feature of it [8, 28].

3.2 Cartesian impedance controller

In order to drive the robot tracking the assembly motion trajectories $\left\{\hat{\xi}_{\mathcal{O}}, \hat{q}\right\}$, the Cartesian impedance controller [9] is utilized in the experiment. With joint torque τ as input, the controller turns out to be

$$\tau = \mathbf{J}^T\left(\mathbf{K}\begin{bmatrix} \hat{\xi}_{\mathcal{O}} - \xi_{\mathcal{O}} \\ \log\left(\hat{q} * \bar{q}\right) \end{bmatrix} - \mathbf{D}\begin{bmatrix} v \\ \omega \end{bmatrix}\right) - \mathbf{J}^T\hat{w} + h(\theta, \dot{\theta}) \tag{10}$$

where

$$\mathbf{K} = \begin{pmatrix} \mathbf{K}^{pos} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}^{ori} \end{pmatrix}, \quad \mathbf{D} = \begin{pmatrix} \mathbf{D}^{pos} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}^{ori} \end{pmatrix}$$

are semi-positive definite matrices of impedance parameters. $\mathbf{J}$ in (10) is the Jacobian and $h(\cdot)$ represents dynamic model of the robot that allows for compensation of gravity, Coriolis force and friction. v and ω stand for linear and angular velocities of the robot's end-effector. $\mathbf{J}^T\hat{w}$ serves as a feed forward term. In the experiment, both $\mathbf{K}$ and $\mathbf{D}$ are set to be diagonal matrices, i.e.

$$K^{pos/ori} = \gamma^{pos/ori}I, \quad D^{pos/ori} = \beta^{pos/ori}I$$

where $\gamma^{pos/ori} > 0$ and $\beta^{pos/ori} > 0$. The control input τ is fed to the robot controller at 1kHz. In the approaching phase, compared with compliance what makes greater sense for the controller is driving the robot to the approaching pose as soon as possible, that is, efficiency dominates. Hence, in this phase, high stiffness ($\gamma^{pos} = 6000$ and $\gamma^{ori} = \sqrt{\gamma^{pos}}$) is assigned to the controller and there is no feed forward term in the controller (10). The assembling phase puts an requirement on the compliance of the robot's end-effector due to unavoidable position or orientation error and tight tolerances between the parts. During the assembling phase, stiff execution may not be safe for robots and objects in interaction [30]. Therefore, low stiffness ($\gamma^{pos} = 3000$ and $\gamma^{ori} = \sqrt{\gamma^{pos}}$) is assigned to the controller in operation to perform compliant assembly which is robust against motion inaccuracy and disturbance [31, 32]. In addition, the damping coefficients are $\beta^{pos} = 100$, $\beta^{ori} = \sqrt{\beta^{pos}}$ for both assembly phases in the experiment.

The n-th reference pose $\left\{\hat{\xi}_{\mathcal{O},n}, \hat{q}_n\right\}$ is fed to the robot controller constantly until the position tracking error $\|\hat{\xi}_{\mathcal{O}} - \xi_{\mathcal{O}}\|$ converges to a given threshold which is 0.0002m in the experiments. Data point in the generated assembly motion trajectory $\left\{\hat{\xi}_{\mathcal{O},n}, \hat{q}_n\right\}_{n=1}^{N}$ are tracked one by one. The impedance parameters of the controller are changed online without halting the motion of the robot.

3.3 Experiment results & discussion

Demonstrations are performed from randomly selected initial pose to an unique pre-assembly and goal pose. In this experiment, rather than absolute time a phase variable $t \in [0, 1]$ is selected as query variable for position data generation. Figure 8 exhibits the TP-GMM learned from the demonstration data via EM algorithm. The assembly polices for the approaching and assembling phases are learned separately and we assign $K = 8$ in (1) for the former one and $K = 5$ for the latter one. The number of Gaussian distributions K is the only hyper-parameter of TP-GMM (1). In this experiment, the hyper-parameters of the assembly policies in the approaching and assembling phase are optimally and individually selected based the Bayesian information criterion (BIC) [33]. Moreover, the demonstrated motion in the assembling phase is much smaller than that in the approaching phase. This partially accounts for the necessity of separating the demonstration and learn two specific assembly policies. Some key motion

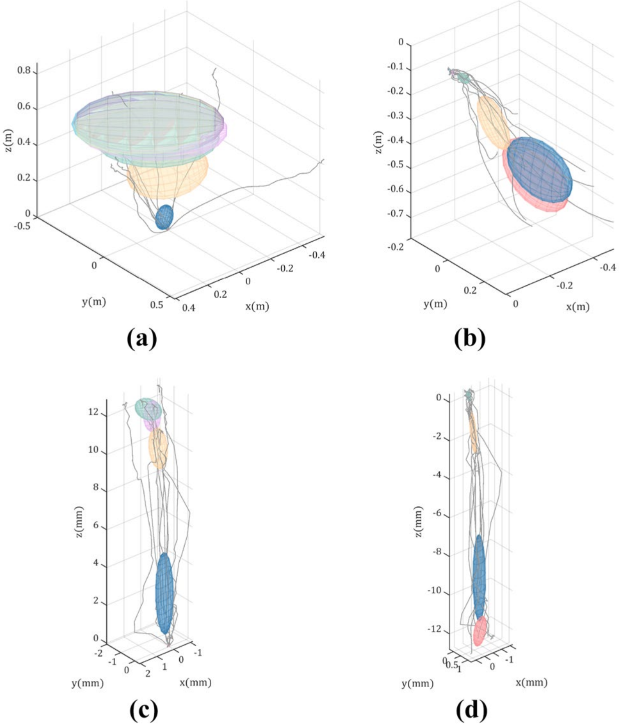
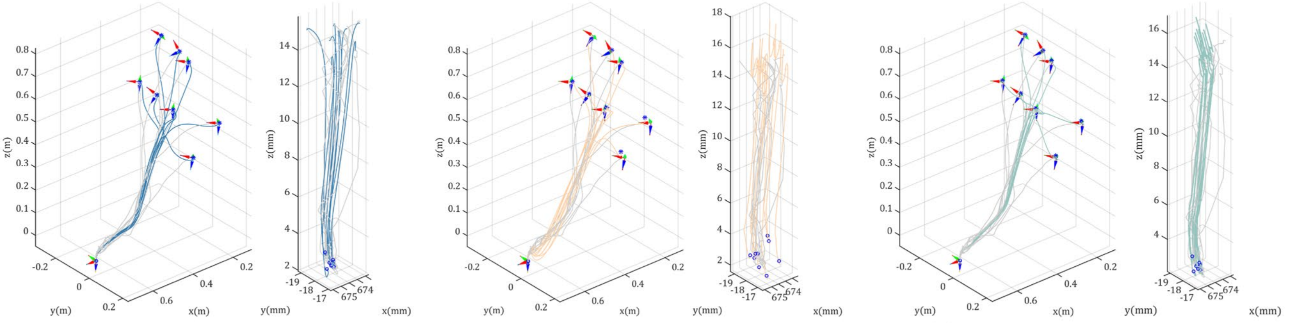


Fig. 8 The learned assembly policy for the approaching phase in the initial frame (**a**) and pre-assembly frame (**b**), and assembly policy for assembling phase in the pre-assembly frame (**c**) and goal frame (**d**). Each ellipsoid represents a Gaussian distribution $\mathcal{N}(\mu^{(j)}_{\xi_{\mathcal{O}},k}, \Sigma^{(j)}_{\xi_{\mathcal{O}},k})$. Coordinates of the centers are the means $\mu^{(j)}_{\xi_{\mathcal{O}},k}$, $k = 1, 2, \ldots, K$. Each ellipsoid has three semi-axes whose directions are along the eigenvectors of the co-variance matrix $\Sigma^{(j)}_{\xi_{\mathcal{O}},k}$ and lengths equal to square roots of the corresponding eigenvalues. The grey curves are demonstration data within the approaching phase (**a**)-(**b**) and assembling phase (**c**)-(**d**). These ellipsoids are drawn in different color to be distinguishable from each other



(a) The proposed policy **(b)** αTP-GMR with $K = 8$ **(c)** αTP-GMR with $K = 13$

Fig. 9 Demonstrated (grey) and retrieved (colored) position trajectories of the whole assembly process (left) and the assembling phase (right) in which "$*$" denotes the starting positions and "$\circ$" denotes the end positions of the retrieved trajectories. The frames indicate the given initial and goal frames

Table 1 Lengths of the generated position trajectories in the demonstration retrieval experiment. The "*Fail*" indicates that end-effector of the robot collided with something in the environment unexpectedly when tracked the trajectory. (Unit: *m*)

Experiment number	1	2	3	4	5	6	7	8	9
Proposed method	0.7022	0.7043	0.9031	0.9066	0.7575	0.8513	1.0352	0.941	0.8503
αTP-GMR ($K = 8$)	0.7338(*Fail*)	0.7403(*Fail*)	0.8876(*Fail*)	0.9043	0.7551(*Fail*)	0.8246	0.9961	0.9241	0.852(*Fail*)
αTP-GMR ($K = 13$)	0.7569	0.7687	0.9457	0.9167	0.8904	0.8845	1.0395	0.9504	0.9262

in the assembling phase may be deemed to be noise once only one policy is learned to model the whole assembly process.

Figure 9 shows both the demonstrated and retrieved position trajectories of the whole assembly process (left) and assembling phase (right). *Retrieval* means generating the trajectories by the learned assembly policies given the same task frames (2) as the demonstrated ones [20]. Figure 9 (a) exhibits that position trajectories are retrieved by the proposed method. For comparison purposes, two TP-GMM [20] based assembly polices, whose $K = 8$ and $K = 13$ respectively, are exploited in the experiment as proposed in [17, 18]. The success rate and length of the generated position trajectory are taken as the performance indices to validate the reliability and efficiency of the proposed LfD method. For the sake of fairness, the orientation and wrench trajectories are also generated through the proposed orientation and wrench policy. And αTP-GMR is also utilized for position trajectories generation, which is different from [17, 18]. Moreover, the K-means algorithm is used to initialize the parameters of the proposed policy and the policies for comparison. As shown by the orange curves in Fig. 9(b), some motion trajectories generated by typical αTP-GMR with $K = 8$ fail to be feasible because some segments of them around the goal pose are *outside* the demonstration data. In the experiment, the PCB may contact with the bottom case in an undesired relative pose once follows those trajectories, which leads to assembly failure or even part damage. Though it has the same number of Gaussian distributions as the proposed approaching policy in the experiment, key positions of the demonstrated assembly are not encapsulated by it. However, when $K = 13$, which equals to sum of that of our approach and assembling policy, TP-GMM with large number of Gaussian distributions can encapsulate not only key positions but also redundant motions in the demonstration data. The redundant motion represents the unnecessary movement of the PCB to be assembled to the bottom case. As indicated by the green curves in Fig. 9(c), there are more redundant motions generated by αTP-GMR with $K = 13$ compared with the blue curves in Fig. 9(a). In the right figures of both Fig. 9(b) and (c), there are *jagged* curves around the beginning of the assembling phase. This is because the demonstration trajectories are jagged caused by the human operator's repeatedly aligning pose of the PCB to the pre-assembly pose during the demonstration process. Lots of redundant demonstration motion are collected and then used to learn the TP-GMM-based policy. On the contrary, with the proposed LfD method those redundant motions are reduced. Table 1 presents the lengths of the retrieved position trajectories to quantitatively indicates the greater reliability and higher efficiency of the proposed LfD method compared with αTP-GMR with $K = 8$ and $K = 13$. The success rate of αTP-GMR with $K = 8$ is the lowest ($< 44.45\%$), which is referred to as the

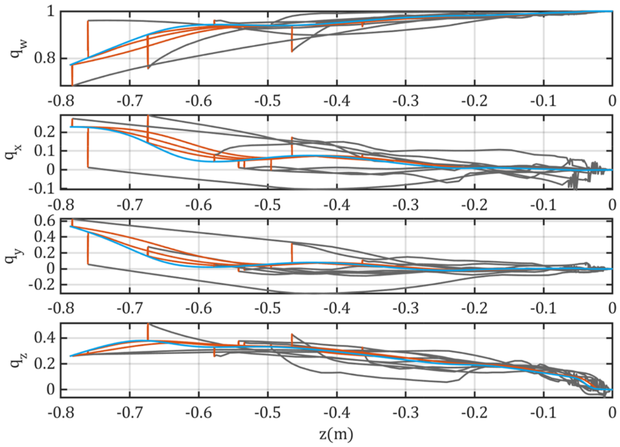


Fig. 10 Demonstrated (grey), retrieved (blue) and recorded (orange) orientation profiles in the approaching phase

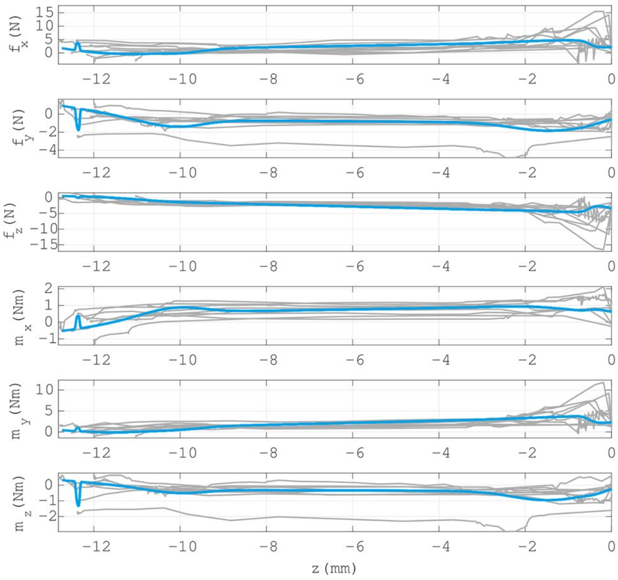
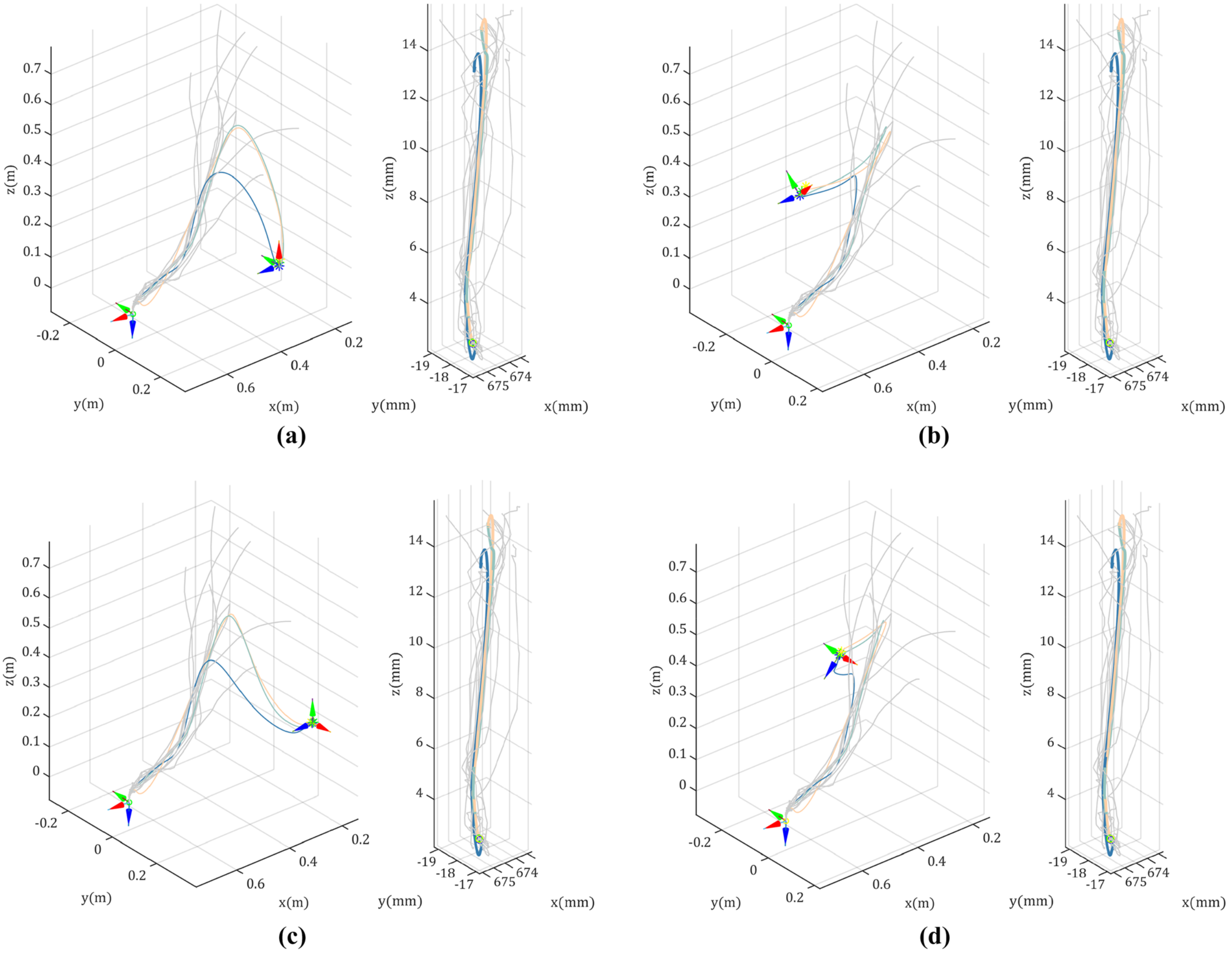


Fig. 11 Demonstrated (grey) and retrieved (blue) wrench profiles in the assembling phase. The recorded wrench profiles are not plotted since the wrench profile is not required to be tracked by the robot

under-fitting problem. Compared with lengths of the trajectories retrieved by αTP-GMR with $K = 13$, those retrieved by the proposed method are shorter in each experiment episode. High efficiency is achieved by the proposed method, which leads to higher productivity in the robotic assembly applications. Note that it is not fair to compare the time consumption of these methods. This is because two impedance controllers of different parameters are used in the proposed method but cannot be used in the two comparison experiments since there is no pre-assembly pose defined therein. In Fig. 9 each initial position generated by αTP-GMR does not coincides with the corresponding initial frame and so does the end position, as a result of which extra control periods are required for the end-effector of the robot to move to the initial pose.

Figure 10 exhibits the demonstrated, retrieved and recorded orientation trajectories in the approaching phase. As queried by the z-axis position in the pre-assembly frame, those trajectories converge to the orientation $[1, 0, 0, 0]^T$ in the pre-assembly frame. Due to the same reason, the retrieved orientation trajectories (blue curves) overlap each other. As shown in Fig. 4, the orientation trajectories are retrieved in advance in this experiment. However, with the proposed coupling strategy, reference orientation can also be queried at any time step given current Cartesian position. This online adaptation technique can deal with unexpected perturbation during the assembly process. Figure 11 exhibits the demonstrated and retrieved wrench profiles in the assembling phase. Only one retrieved wrench profile is exhibited because the parts to be assembled do not vary among the experiment scenarios. As discussed in Section 2.2, we do not drive the robot tracking the wrench trajectory as it only serves as a feed forward term and safety indicator.

To verify the generalization capability of our proposed method, 4 novel initial poses are selected in the experiment.



(a)
(b)
(c)
(d)

Fig. 12 Demonstrated (grey) and generalized (colored) position data of the whole assembly process (left) and the assembling phase (right) where the blue one is generated by the proposed method, the orange one is generated by αTP-GMR with $K = 8$, and the green one is generated by αTP-GMR with $K = 13$. "$*$" denotes the starting positions and "$\circ$" denotes the goal positions of the assembly motion trajectories. The frames indicate the given initial and goal frames

Table 2 Lengths of the generated position trajectories in the initial pose generalization experiment. (Unit: *m*)

Experiment number	1	2	3	4
Proposed method	0.8914	0.846	0.9149	0.8426
αTP-GMR ($K = 8$)	1.118	1.1525	1.1006	1.0866
αTP-GMR ($K = 13$)	1.1251	1.1505	1.096	1.0772

By *generalization* what we mean is to generate the trajectories by the learned policy given the task frames (2) outside the demonstration ones [20]. Here only the initial frames are changed in each experiment episode since the initial pose of the gripped part (PCB) may vary but the goal pose is kept fixed in the robotic assembly tasks as exhibited by Fig. 1. Figure 12 shows the position trajectories generated by the proposed LfD method (blue), αTP-GMR with $K = 8$ (orange) and αTP-GMR with $K = 13$ (green). The position trajectories generated by the proposed method contain less redundant motions than those generated by the other two policies in each experiment episode. Table 2 presents lengths of generalized trajectories, among which the ones generated by the proposed policy are the shortest in each experiment episode. Some segments of the orange curves in Fig. 12 are also outside the demonstration data around the pre-assembly pose. Figure 13 presents the generalized orientation trajectories (blue curves) via GMR and the recorded orientation trajectories during assembly process (orange curves). In contrast to Fig. 10, orientation data in Fig. 13 are plotted along the time sequence in the approaching phase due to the fact that variable z is not monotonously varying in these cases but the orientation policy (7) still works. In Figs. 10 and 13, it can be seen that during the approaching phase the recorded orientation trajectories quickly converge to the retrieved/generalized ones almost before the motion of position. This is mainly because the absolute value of the initial orientation error $\log(\widehat{q}(0) * \bar{q}(0))$ in (10) is numerically much larger than that of position.

In the generalization experiment, only the initial frame varies. We can also change the goal frame by moving the mouse bottom case to another pose which will be estimated by the 3D camera as shown in Fig. 7. Anyway, the demonstration data within the approaching phase can only provide limited information about the environment, which means that unexpected collision with some objects, such as the holder of the 3D camera, may occur during the approaching phase. With TP-GMM-based policy, this problem can be solved by extra demonstration or correction demonstration from the human operator to adjust the parameters of the policy [33].

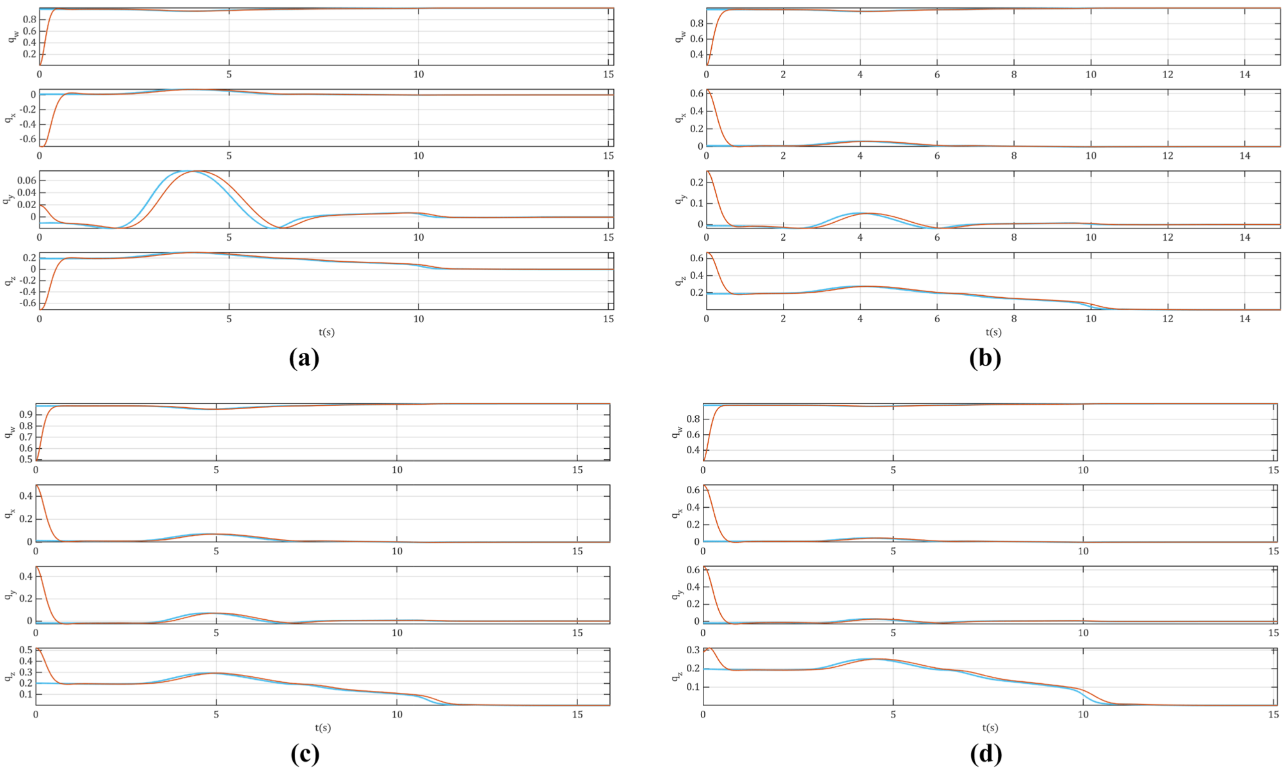


Fig. 13 Orientation trajectories generated by the orientation policy (blue) and the recorded orientation trajectories (orange) during the approaching phase

4 Conclusion

To enable robots quickly acquire new assembly skills in assembly lines, the learning from demonstration paradigm provides a flexible solution to transfer workers' assembly skills to robots. A robot LfD framework is developed in this article for skillful industrial assembly applications. In this framework, the assembly task is divided into approaching and assembling phases, which is the key to avoid the potential under-fitting problem introduced by the large difference in variability of demonstration data between the two assembly phases. By exploiting the stochastic nature and generalization capability of TP-GMM, the demonstrated position data are modeled and retrieved through αTP-GMR. The corresponding orientation and wrench are also queried by the position via GMM, which turns out to be an effective coupling strategy that enables online adaptation of orientation and wrench. A PCB assembly experiment is taken as illustration for the proposed robot LfD assembly framework. It indicates that the learned policies can generate feasible and efficient assembly position and orientation trajectories in both phases.

Author contributions These authors contributed equally to this work.

Funding information This work was supported by NSFC-Shenzhen Robotics Basic Research Center Program Grant U1713202.

Availability of data and materials Not applicable.

Declarations

Ethical approval Not applicable.

Consent to participate We consent.

Consent to publish We consent.

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Cohen Y, Naseraldin H, Chaudhuri A, Pilati F (2019) Assembly systems in industry 4.0 era: a road map to understand assembly 4.0. Int J Adv Manuf Tech 105(9):4037–4054
2. Pedersen MR, Nalpantidis L, Andersen RS, Schou C, Bøgh S, Krüger V, Madsen O (2016) Robot skills for manufacturing: From concept to industrial deployment. Robotics and Computer-Integrated Manufacturing 37:282–291
3. Qu Y, Ming X, Liu Z, Zhang X, Hou Z (2019) Smart manufacturing systems: state of the art and future trends. Int J Adv Manuf Tech 103(9):3751–3768
4. Sloth C, Kramberger A, Iturrate I (2020) Towards easy setup of robotic assembly tasks. Adv Robot 34(7–8):499–513
5. Wang Y, Xiong R, Yu H, Zhang J, Liu Y (2018) Perception of Demonstration for Automatic Programing of Robotic Assembly: Framework, Algorithm, and Validation. IEEE/ASME Trans Mechatron 23(3):1059–1070
6. Ravichandar H, Polydoros AS, Chernova S, Billard A (2020) Recent Advances in Robot Learning from Demonstration. Annual Review of Control, Robotics, and Autonomous Systems 3(1):297–330
7. Gašpar T, Deniša M, Radanovič P, Ridge B, Savarimuthu TR, Kramberger A, Priggemeyer M, Roßmann J, Wörgötter F, Ivanovska T, Parizi S, Gosar Ž, Kovač I, Ude A (2020) Smart hardware integration with advanced robot programming technologies for efficient reconfiguration of robot workcells. Robotics and Computer-Integrated Manufacturing 66
8. Wang Y, Harada K, Wan W (2020) Motion planning of skillful motions in assembly process through human demonstration. Adv Robot
9. Zeng C, Yang C, Cheng H, Li Y, Dai SL (2020) Simultaneously Encoding Movement and sEMG-based Stiffness for Robotic Skill Learning. IEEE Trans Industr Inf pp. 1–1
10. Kramberger A, Gams A, Nemec B, Chrysostomou D, Madsen O, Ude A (2017) Generalization of orientation trajectories and force-torque profiles for robotic assembly. Robot Auton Syst 98:333–346
11. Kwak SJ, Hasegawa T, Mozos OM, Chung SY (2014) Elimination of unnecessary contact states in contact state graphs for robotic assembly tasks. Int J Adv Manuf Tech 70(9–12):1683–1697
12. Park H, Park J, Lee DH, Park JH, Baeg MH, Bae JH (2017) Compliance-Based Robotic Peg-in-Hole Assembly Strategy Without Force Feedback. IEEE Trans Industr Electron 64(8):6299–6309
13. Ehlers D, Suomalainen M, Lundell J, Kyrki V (2019) Imitating Human Search Strategies for Assembly. In: 2019 International Conference on Robotics and Automation (ICRA), pp. 7821–7827
14. Fan Y, Luo J, Tomizuka M (2019) A Learning Framework for High Precision Industrial Assembly. In: 2019 International Conference on Robotics and Automation (ICRA), pp. 811–817
15. Inoue T, De Magistris G, Munawar A, Yokoya T, Tachibana R (2017) Deep reinforcement learning for high precision assembly tasks. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 819–825
16. Thomas G, Chien M, Tamar A, Ojea JA, Abbeel P (2018) Learning robotic assembly from cad. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 3524–3531. IEEE
17. Duque DA, Prieto FA, Hoyos JG (2019) Trajectory generation for robotic assembly operations using learning by demonstration. Robotics and Computer-Integrated Manufacturing 57:292–302
18. Kyrarini M, Haseeb MA, Ristić-Durrant D, Gräser A (2019) Robot learning of industrial assembly task via human demonstrations. Auton Robot 43(1):239–257
19. Sena A, Michael B, Howard M (2019) Improving Task-Parameterised Movement Learning Generalisation with Frame-Weighted Trajectory Generation. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4281–4287
20. Calinon S (2016) A tutorial on task-parameterized movement learning and retrieval. Intel Serv Robot 9(1):1–29
21. Huang Y, Abu-Dakka FJ, Silvério J, Caldwell DG (2019) Generalized Orientation Learning in Robot Task Space. In: 2019 International Conference on Robotics and Automation (ICRA), pp. 2531–2537
22. Calinon S (2020) Gaussians on Riemannian Manifolds: Applications for Robot Learning and Adaptive Control. IEEE Robotics Automation Magazine 27(2):33–45
23. Morais JP, Georgiev S, Sprößig W (2014) Real quaternionic calculus handbook. Springer

24. Zeestraten MJA, Havoutis I, Silvério J, Calinon S, Caldwell DG (2017) An Approach for Imitation Learning on Riemannian Manifolds. IEEE Robotics and Automation Letters 2(3):1240–1247
25. Rozo L, Calinon S, Caldwell DG, Jiménez P, Torras C (2016) Learning Physical Collaborative Robot Behaviors From Human Demonstrations. IEEE Trans Rob 32(3):513–527
26. Andrzejewski K, Cooper M, Griffiths C, Giannetti C (2018) Optimisation process for robotic assembly of electronic components. Int J Adv Manuf Tech 99(9):2523–2535
27. Markley FL, Cheng Y, Crassidis JL, Oshman Y (2007) Averaging quaternions. J Guid Control Dyn 30(4):1193–1197
28. Wierschem DC, Jimenez JA, Mediavilla FAM (2020) A motion capture system for the study of human manufacturing repetitive motions. Int J Adv Manuf Tech 110(3):813–827
29. Yang C, Zeng C, Cong Y, Wang N, Wang M (2019) A Learning Framework of Adaptive Manipulative Skills From Human to Robot. IEEE Trans Industr Inf 15(2):1153–1161
30. Petrič T, Gams A, Colasanto L, Ijspeert AJ, Ude A (2018) Accelerated Sensorimotor Learning of Compliant Movement Primitives. IEEE Trans Rob 34(6):1636–1642
31. Abu-Dakka FJ, Rozo L, Caldwell DG (2018) Force-based variable impedance learning for robotic manipulation. Robot Auton Syst 109:156–167
32. Wirnshofer F, Schmitt PS, Meister P, Wichert G, Burgard W (2019) Robust, Compliant Assembly with Elastic Parts and Model Uncertainty. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6044–6051
33. Cao Z, Hu H, Zhao Z, Lou Y (2019) Robot Programming by Demonstration with Local Human Correction for Assembly. In: 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 166–171