Efficient Insertion Control for Precision Assembly Based on Demonstration Learning and Reinforcement Learning

Yanqin Ma, De Xu, *Senior Member*, *IEEE* and Fangbo Qin

***Abstract*—Multiple peg-in-hole insertion control is one of the challenging tasks in precision assembly for its complex contact dynamics. In this work, an insertion policy learning method is proposed for multiple peg-in-hole precision assembly. The insertion policy learning process is separated into two phases: initial policy learning and residual policy learning. In initial policy learning, a state-to-action policy mapping model based on Gaussian mixture model (GMM) is established. And Gaussian mixture regression (GMR) is used to generalize the policy reuse. In residual policy learning, a reinforcement learning method named normalized advantage function (NAF) is employed to refine the insertion policy via agent's exploration in insertion environment. Moreover, an adaptive action exploration strategy is designed to improve the performance of exploration, and the prioritized experience replay strategy is introduced to make the residual policy learning from historical experience more efficient. Besides, the hierarchical reward function is designed considering the contact dynamics as well as the efficiency and safety of precision insertion. Finally, comprehensive experiments are conducted to validate the effectiveness of the proposed insertion policy learning method.**

***Index Terms*—Precision assembly, multiple peg-in-hole insertion, insertion policy learning, demonstration learning, reinforcement learning**

I. Introduction

PRECISION assembly is one of key technologies in manufacture fields, and it is widely applied in micro-electro-mechanical system, electronic engineering, bioengineering and aerospace [1][2]. Precision assembly refers to manipulating small-sized objects with high accuracy [1]. In general, precision assembly consists of two phases: alignment and insertion. In alignment phase, image-based visual servo control methods are commonly employed to align objects' orientations and positions since microscopic vision can provide accurate and non-contact information [1]-[4]. However, in insertion phase, the force-based control methods are usually considered since the objects are mutual blocked from vision and these small objects are prone to be damaged in case of large contact forces. In particular, it is necessary to design an appropriate radial contact forces regulation mechanism in interference assembly to protect objects from damage [4].

In general, traditional force-based insertion control methods are classified into model-free methods [5] and model-based methods [6]. Model-free methods employ active force feedback to accomplish compliant insertion, which is time-consuming and tedious to obtain perfect controller's parameters via trial and error [5]. Therefore, it is more practical to establish a contact dynamics model for precision insertion and design model-based insertion control method [6]. Based on Hooke's law of elasticity, Liu *et al.* [7] developed a linearized insertion dynamic model, which achieves precision insertion assembly of two cylindrical objects. However, the stability of the designed control system can be guaranteed within a small range near the equilibrium. Moreover, it is difficult to establish a precise model due to the uncertainty and noise in real-world insertion system. Particularly, the contact dynamics of multiple peg-in-hole precision insertion is more complex [8]. The insertion policies without learning require human to design them based on the experience. Most of the aforementioned insertion algorithms are based on specific characteristics of insertion task, which has less adaptability and generality. Therefore, it is necessary to develop an efficient insertion control method, which not only has robustness to disturbances in insertion process, but also has generalization to different insertion tasks.

Different from manual programming without learning, which requires a great deal of deep analysis of the insertion task, human can dexterously complete insertion task with their experience, i.e., learned skill. Besides, the merits of human assembly, such as flexibility, impedance feature and variable stiffness regulation are considered, which are important to facilitate task's performance [9]. Therefore, introducing human skill learning to precision insertion is an appealing approach to solve the exiting problems [10]. Recently, typical skill learning methods include reinforcement learning methods [11]-[15] and demonstration learning methods [16]-[18].

In reinforcement learning, the agent directly learns behaviors from interacting with environment without any assumptions on the system, and an optimal policy can be obtained through sufficient exploration under the guidance of the reward function. Reinforcement learning has been widely used in insertion task. Inoue *et al.* [11] used a long short term memory

Manuscript received February 24, 2020; revised May 08, 2020 and June 24, 2020; accepted August 10, 2020. This work is supported in part by the National Natural Science Foundation of China under Grants 61873266 and 61733004. *(Corresponding author: De Xu).*

Y. Ma is with the College of Automation, Nanjing Institute of Technology, Nanjing 211167, China, and also with the Research Center of Precision Sensing and Control, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: mayanqin2017@ia.ac.cn).

D. Xu and F. Qin are with the Research Center of Precision Sensing and Control, Institute of Automation, Chinese Academy of Sciences and University of Chinese Academy of Sciences, Beijing 101408, China (e-mail: de.xu@ia.ac.cn; qinfangbo2013@ia.ac.cn).

(LSTM) network to learn high precision peg-in-hole assembly policy. It needs to discretize the action space to a finite number of actions, which decreases the flexibility of continuous insertion tasks. Luo *et al.* [12] proposed a model-based reinforcement learning method, i.e., mirror decent guided policy search (MDGPS), to assemble rigid object to deformable hole. Xu *et al.* [13] combined deep deterministic policy gradient (DDPG) and force feedback to accomplish multiple peg-in-hole insertion. These mentioned insertion skill learning methods show impressive performance. However, it is unsafe when the agent directly explores in real-world environment. Meanwhile, the time and labor costs are extremely high to learn an optimal policy. Although via transferring learned policy from simulation to real-world may increase learning efficiency and improve safety, it is difficult to guarantee the realism of simulation due to complex contact dynamics in insertion [15].

Demonstration learning refers to reusing and generalizing the learned skill from an expert or agent to another agent by imitating demonstration trajectory. Demonstration learning is a common way to acquire new skill and widely investigated [16]. Wang *et al.* [17] developed a demonstration perception system for robotic assembly, in which a probabilistic framework for system modeling, a multi-model information fusion strategy for action recognition, and an expectation maximization method for demonstration skill generation are combined. Ehlers *et al.* [18] proposed an insertion searching control framework based on human demonstrations, which performs well both in 2D peg-in-hole and 3D electricity socket insertion tasks. Demonstration learning is data-efficiency, but it is difficult to guarantee the learned insertion policy is global optimal due to the finite demonstration space. Particularly, for multiple peg-in-hole precision insertion task which has rich contact dynamics, it is difficult to learn an optimal insertion policy only from demonstration learning.

In general, for precision insertion with complex contact dynamics, combining reinforcement learning with demonstration learning is a powerful approach [19][20]. On the one hand, demonstration learning can help reinforcement learning to quickly obtain useful information for exploration in environment. On the other hand, the prior knowledge from demonstrations can be integrated into reinforcement learning to accelerate learning and keep learning steady [21][22]. It not only takes the efficiency of demonstration learning but also the flexibility of reinforcement learning [23].

Inspired by these researches, we propose an efficient insertion policy learning method to achieve multiple peg-in-hole precision insertion. Different from traditional methods which require a large amount of demonstration data to pre-train the insertion policy and fine-tune the insertion policy through reinforcement learning, the proposed method separates the insertion policy into initial policy and residual policy [24]. The initial policy learning is based on the established policy mapping model, and the initial policy can be well learned from several demonstrations off-line since the established policy mapping model has good generalization ability. Considering multiple peg-in-hole insertion is a complex continuous control problem, a modified normalized advantage function is well

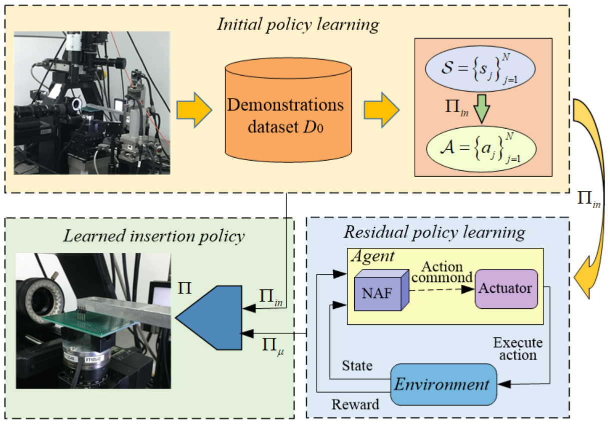


Fig. 1. Framework for precision insertion policy learning.

designed to learn residual policy, and the learning efficiency is effectively improved. The main contributions of this paper are as follows.

(1) An efficient policy learning method for multiple peg-in-hole precision insertion is proposed, which consists of initial policy learning and residual policy learning. A policy mapping model based on GMM is developed to represent the relation from states to actions. Then the initial policy can be obtained by generalizing the policy mapping model with GMR. Besides, a modified normalized advantage function is well developed to learn the residual policy efficiently and steady.

(2) In residual policy learning, an adaptive action exploration strategy based on one-step reward increment is designed to improve the efficiency of residual policy learning. Besides, a prioritized experience replay strategy based on temporal difference (TD) error is also introduced to make the collected insertion experience replay more efficient.

The rest of this paper is organized as follows. Section II presents the preliminaries of the proposed method. Section III presents the insertion policy learning method in detail. Precision insertion experiments are carried out in section IV. Finally, conclusions are presented in section V.

II. Preliminaries

A. Problem Statement

The precision insertion of multiple peg-in-hole can be described by a dynamical system [16]

$$\dot{s} = g(s,a) \tag{1}$$

where $s=[f_x, f_y, f_z]^T$ represents system's state, i.e., the contact forces captured by the force sensor, $a=[d_x, d_y, d_z]^T$ represents system's input, i.e., the manipulator's action including the translational movements along axes X, Y and Z.

To complete precision insertion, i.e., make the system's state transfer from initial state s_0 to desired state s^*, it is necessary to calculate the action a for a given state s [7]. However, it is difficult to obtain the precise analytical form of the dynamical system (1) due to the complex contact dynamics in precision insertion. Therefore, we assume the action for system (1) at given state s is

$$a = \Pi(s) \tag{2}$$

where $\Pi(\cdot)$ represents the insertion policy, it is a mapping func-

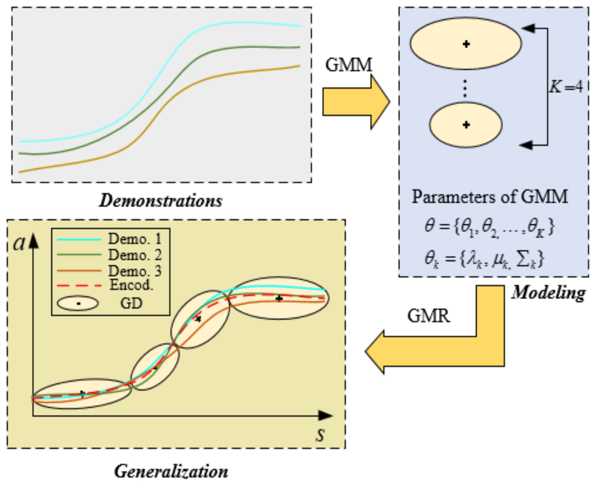


Fig. 2. Framework of initial policy learning.

tion from state s to action a.

The goal of the insertion policy learning is

$$\min \sum_{i=1}^{\xi} \left\| \Pi(s_i) - \Pi^*(s_i) \right\| \tag{3}$$

where ξ is the number of states in precision insertion, and $\Pi^*(\cdot)$ is the optimal insertion policy.

B. Reinforcement Learning

Markov decision process is a stochastic process satisfying Markov property, which mainly consists of four components: the state space $\mathcal{S}$, the action space $\mathcal{A}$, the reward function $\mathcal{R}$ and the transition probability $p(s_{t+1}|s_t, a_t)$. These components are necessary in reinforcement learning.

In reinforcement learning, the purpose is to learn an optimal policy via maximizing the expected accumulated reward. At each time-step t of an episode, the agent interacts with environment and conducts action a_t based on the current insertion policy Π. Then the state s_t is transferred to the next state s_{t+1}, and the agent receives a reward $r_t=r(s_t, a_t)$. We define an action value function $Q^{\Pi}(s_t,a_t)=\mathbb{E}\left[\sum_{i=t}^{T}\gamma^{i-t}r_i\right]$ with discount factor $\gamma\in[0,1]$ and the horizon of the episode T, then the optimal policy Π^* can be learned via maximizing the following action value function.

$$\max_{a_t\in\mathcal{A}} Q^{\Pi}(s_t,a_t) = \mathbb{E}^{\Pi}\left[r_t + \gamma \max_{a_{t+1}\in\mathcal{A}} Q^{\Pi}(s_{t+1},a_{t+1})\right] \tag{4}$$

Equation (4) is a recursive function, i.e., *Bellman equation*.

III. Insertion Policy Learning

The overall learning framework for precision insertion is shown in Fig. 1, which combines demonstration learning to learn initial policy and reinforcement learning to learn residual policy. The insertion policy $\Pi(s)=\Pi_{in}(s)+\Pi_{\mu}(s)$ can be obtained once the initial policy $a^{in}=\Pi_{in}(s)$ and the residual policy $a^{\mu}=\Pi_{\mu}(s)$ are determined, separately. GMM/GMR and NAF are two main elements in the proposed insertion policy learning method. GMM can effectively synthesize the underlying common features of multiple demonstrations, and GMR can generalize the learned insertion policy from only several demonstrations. The residual policy is learned based on modified NAF which is a classic reinforcement learning method for continuous action controlling. The learned initial policy can accelerate residual policy learning. In specific, if the initial policy is nearly perfect, the residual policy plays a corrective role. However, if the initial policy is far from the optimal insertion policy, the residual policy optimizes the insertion policy with the guidance of the initial policy.

A. Initial Policy Learning

The policy mapping model is developed in this section to make the agent learn the initial policy. Firstly, the model is built based on GMM and the model's parameters are learned from demonstration data. Then the generalization of the initial policy is deduced from GMR.

1) Policy Mapping Model

Taking 1-D mapping with four Gaussian distributions encoding three demonstrations as an example, the framework of the designed policy mapping model is shown in Fig. 2. The structure and learning of the model are introduced in detail.

GMM: Assume K Gaussian distributions are needed to estimate the initial policy $\Pi_{in}(s)$. Thus the estimation of $\Pi_{in}(s)$ becomes learning the parameter $\theta=\{\theta_1, \theta_2,\dots, \theta_K\}$ with $\theta_k=\{\lambda_k, \mu_k, \Sigma_k\}$. The mixture coefficient λ_k, the mean matrix μ_k and the covariance matrix Σ_k are parameters of k-th Gaussian distribution. Particularly, we define the mean and covariance matrices of k-th Gaussian distribution are

$$\mu_k = \begin{bmatrix} \mu_k^{a^{in}} \\ \mu_k^{s} \end{bmatrix} \quad \Sigma_k = \begin{bmatrix} \Sigma_k^{a^{in}} & \Sigma_k^{a^{in},s} \\ \Sigma_k^{s,a^{in}} & \Sigma_k^{s} \end{bmatrix} \tag{5}$$

The joint probability density of a^{in} and s modeled by the mixture of K Gaussian distributions yields

$$p\left(\left[a^{in},s\right]\middle|\theta\right) = \sum_{k=1}^{K} \lambda_k\, p\left(\left[a^{in},s\right]\middle|\mu_k,\Sigma_k\right) \tag{6}$$

where the conditional probability density of k-th Gaussian distribution $p([a^{in}, s]|\mu_k, \Sigma_k)$ is

$$\begin{aligned} p\left(\left[a^{in},s\right]\middle|\mu_k,\Sigma_k\right) &= \mathcal{N}\left(\left[a^{in},s\right]\middle|\mu_k,\Sigma_k\right) \\ &= \frac{\exp\left[-0.5\left(\left[a^{in},s\right]-\mu_k\right)^{\mathrm{T}}\left(\Sigma_k\right)^{-1}\left(\left[a^{in},s\right]-\mu_k\right)\right]}{\sqrt{(2\pi)^{(d_s+d_a)}\left|\Sigma_k\right|}} \end{aligned} \tag{7}$$

where d_s is the length of state, d_a is the length of action, d_s and d_a are integer, $\mathcal{N}$ stands for Gaussian distribution.

Parameter Estimation: To learn the model's parameter θ, the demonstrations are conducted and the demonstration data is stored in the designed dataset $D_0=\{s^j, a^j\}^{j=1:N}$. Expectation–maximization (EM) algorithm with the following optimization function is used to determine the optimal value θ

$$\begin{gathered} \max_{\theta} J(\theta) = \max_{\theta} \frac{1}{N}\sum_{j=1}^{N} \log\left\{\sum_{k=1}^{K}\lambda_k\, p\left(\left[a^{in},s\right]\middle|\mu_k,\Sigma_k\right)\right\} \\ s.t.\ \ \sum_{k=1}^{K}\lambda_k = 1,\ \ 0\le\lambda_k\le 1 \end{gathered} \tag{8}$$

Parameter Initialization: EM algorithm is a local search technique, which is sensitive to initial value. Therefore, to avoid parameter θ getting trapped into a local minimum, k-means clustering algorithm is employed to initialize θ. This algorithm divides the dataset D_0 into multiple sub-sets and aims to find a partition $D_0=\{D_1, D_2, \dots, D_K\}$ via minimizing the following optimization function

$$\min_D \Psi(D)=\sum_{k=1}^{K}\sum_{x\in D_k}\left\|x-m_k\right\|^2 \tag{9}$$

where $x=[s^j, a^j]^T$ represents the data in dataset D_0, m_k is the mean of the data in dataset D_k.

With (9), the initial value of parameter θ for EM algorithm is

$$\lambda_k=|D_k|\Big/\sum_{k=1}^{K}|D_k|,\quad \mu_k=m_k,\quad \Sigma_k=\begin{bmatrix}\Sigma_k^{a^{in}} & \Sigma_k^{a^{in},\ s}\\ \Sigma_k^{s,\ a^{in}} & \Sigma_k^{s}\end{bmatrix}_{x\in D_k} \tag{10}$$

Model Selection: The number of Gaussian distributions will affect the fitting performance of GMM. We use the Bayesian Information Criterion (BIC) to determine the optimal number of Gaussian distributions K. The BIC determines a tradeoff between optimizing mixture model's average log-likelihood and the number of parameters that are needed to encode the data. K can be calculated via minimizing the designed BIC

$$BIC=n_p\log(N)-2J^*(\theta) \tag{11}$$

where N is the number of data points in dataset D_0, $J^*(\theta)$ is the maximized value of likelihood function for the estimated model, as is defined in (8), n_p is the number of free parameters to be estimated, and K can be calculated via minimizing (11).

2) *Generalization for Policy Reuse*

Once the GMM's parameter θ is determined, the general form of the policy mapping model can be reconstructed. Firstly, the conditional probability distribution of action a^{in} for a given state s can be calculated.

$$p\left(a^{in}|s\right)=\frac{p\left(\left[a^{in},s\right]|\theta\right)}{\int p\left(\left[a^{in},s\right]|\theta\right)da^{in}}=\sum_{k=1}^{K}h_k(s)\mathcal{N}\left(a^{in}\middle|\hat{\mu}_k^{a^{in}},\hat{\Sigma}_k^{a^{in}}\right) \tag{12}$$

where

$$h_k(s)=\frac{\lambda_k\mathcal{N}\left(s\middle|\mu_k^s,\Sigma_k^s\right)}{\sum_{i=1}^{K}\lambda_i\mathcal{N}\left(s\middle|\mu_i^s,\Sigma_i^s\right)} \tag{13}$$

$$\begin{aligned}\hat{\mu}_k^{a^{in}}&=\mu_k^{a^{in}}+\Sigma_k^{a^{in},s}\left(\Sigma_k^s\right)^{-1}\left(s-\mu_k^s\right)\\ \hat{\Sigma}_k^{a^{in}}&=\Sigma_k^{a^{in}}-\Sigma_k^{a^{in},s}\left(\Sigma_k^s\right)^{-1}\Sigma_k^{s,a^{in}}\end{aligned} \tag{14}$$

Then the expectation of action a^{in} based on state s gives the estimation of $\Pi_{in}(s)$.

$$a^{in}=\Pi_{in}(s)=\sum_{k=1}^{K}h_k(s)\hat{\Sigma}_k^{a^{in}} \tag{15}$$

With the designed policy mapping model, the initial policy a^{in} for given state s can be calculated from (15).

B. *Residual Policy Learning*

The residual policy learning framework is shown in Fig. 3. The framework is based on modified NAF. In this framework, the evaluation network directly outputs action-value function and residual policy. The target network has the same structure with the evaluation network, which is used to stabilize residual policy learning. Action exploration strategy based on one-step reward increment is designed to increase learning efficiency. Hierarchical reward function is developed to balance the safety and efficiency. Besides, the experience memory R is designed to store experience data (s, a, r, s'), and prioritized experience replay with TD error is developed to improve the replay efficiency of historical experience.

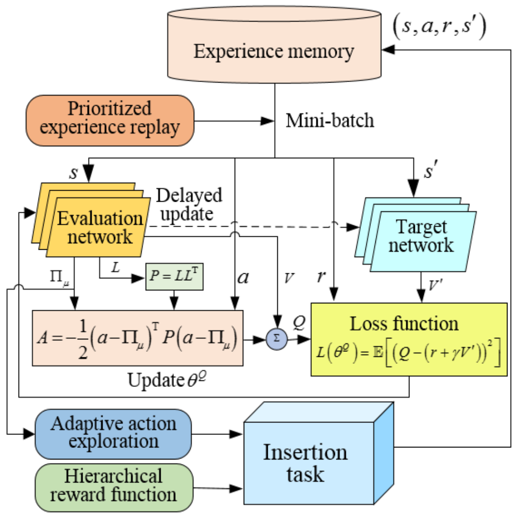


Fig. 3. Framework for residual policy learning.

1) *Training of Networks*

The evaluation network with parameter $\theta^Q=\{\theta^\mu, \theta^P, \theta^V\}$ is used to estimate the action-value function $Q(s_t, a_t|\theta^Q)$, while the target network with parameter $\theta^{Q'}$ is used to compute the target value. The loss function used to train the evaluation network is

$$L(\theta^Q)=\mathbb{E}\left[\left(Q\left(s_t,a_t\middle|\theta^Q\right)-y_t\right)^2\right] \tag{16}$$

where y_t yields

$$y_t=r_t+\gamma V'\left(s_{t+1}\middle|\theta^{Q'}\right) \tag{17}$$

where γ is discount factor, $V'(s_{t+1}|\theta^{Q'})$ is the target value function, which is the output of target network, r_t is one-step reward, which is designed in the following sub-section.

The output of evaluation network $Q(s_t, a_t|\theta^Q)$ is separated into value function $V(s_t|\theta^V)$ and advantage function $A(s_t,a_t|\theta^A)$.

$$\begin{aligned}&Q\left(s_t,a_t\middle|\theta^Q\right)=V\left(s_t\middle|\theta^V\right)+A\left(s_t,a_t\middle|\theta^A\right)\\ &A\left(s_t,a_t\middle|\theta^A\right)=-\frac{1}{2}\left(a_t-\Pi_\mu\left(s_t\middle|\theta^\mu\right)\right)^T P\left(s_t\middle|\theta^P\right)\left(a_t-\Pi_\mu\left(s_t\middle|\theta^\mu\right)\right)\end{aligned} \tag{18}$$

where $P(s_t|\theta^P)$ is a positive definite square matrix, which can be written as $P(s_t|\theta^P)=L(s_t|\theta^P)L(s_t|\theta^P)^T$, $L(s_t|\theta^P)$ is a lower triangular matrix and noted as L.

Differentiating the loss function (16) with respect to parameter θ^Q, the gradient to update θ^Q is obtained.

$$\nabla_{\theta^Q}L(\theta^Q)=\mathbb{E}\left[\left(Q\left(s_t,a_t\middle|\theta^Q\right)-y_t\right)\nabla_{\theta^Q}Q\left(s_t,a_t\middle|\theta^Q\right)\right] \tag{19}$$

To make residual policy's learning steady, the target network with parameter $\theta^{Q'}$ is updated by

$$\theta^{Q'}=\tau\theta^Q+(1-\tau)\theta^{Q'} \tag{20}$$

where τ is the delayed update factor with $\tau<<1$.

2) *Hierarchical Reward Function*

The reward function is used to evaluate the selected action on given state, which is a key component in reinforcement learning. In precision insertion, it is expected to improve the safety and efficiency of insertion task, i.e., to decrease the number of insertion steps as small as possible and to make the radial contact forces f_x and f_y as low as possible. Therefore, we design the following hierarchical reward function to satisfy the requirements of insertion task.

$$r=\begin{cases} r_1 & \text{at the end of each episode} \\ r_2 & \text{for each action in episode} \end{cases} \tag{21}$$

where r_1 is used to evaluate the number of insertion steps, r_1 is calculated at the end of each episode and used as the result of hierarchical reward function. r_2 is used to evaluate the performance of the action. When the episode is in progress, r_2 is calculated after each action and used as the result of hierarchical reward function.

The reward r_1 is calculated

$$r_1=1-k/k_{\max} \tag{22}$$

where $k\in(0, k_{\max})$ is the number of insertion steps in one episode, $k_{\max}$ is the maximum number of insertion steps.

We design a fuzzy reward function to evaluate each action in insertion process. The calculation process of the fuzzy reward mainly includes fuzzifer, fuzzy inference and defuzzifier. In fuzzifer, the inputs $f_r=\|f_x, f_y\|_2$ and axial feeding step length d_z as well as the output r_2 are converted into fuzzy values with triangular membership functions. In fuzzy inference, the Mamdani inference method is used. In defuzzifer, the weighted average function is employed. The fuzzy reward r_2 yields

$$r_2=\Phi\left(f_r, d_z\right)=\sum_{i=1}^{m} R_i\zeta_i \Big/ \sum_{i=1}^{m}\zeta_i \tag{23}$$

where m is the number of rules satisfied, ζ_i is the membership, R_i is the *i-th* rule.

3) *Adaptive Action Exploration*

Different from classic action exploration strategies, such as random Gaussian noise and Ornstein-Uhlenbeck noise, we design an adaptive action exploration strategy to improve the exploration efficiency and prevent the agent from unsafe action exploration for safety.

An adaptive action exploration strategy is designed.

$$a^{\mu}=\Pi_{\mu}\left(s\middle|\theta^{\mu}\right)+\mathcal{N}\left(\delta,\sigma^2\mathrm{I}\right) \tag{24}$$

where Gaussian distribution $\mathcal{N}$ is controlled by δ and σ. To make the action exploration more efficient, δ and σ are tuned after each action according to the incremental reward Δr_t

$$\Delta r_t=r_t-r_t^{in} \tag{25}$$

where r_t is the one-step reward corresponding to action with action exploration $a_t=a_t^{\mu}+a_t^{in}$ based on (24) and (15), which can be calculated from (21), r_t^{in} is the reward corresponding to the action without exploration.

During insertion process, because the executed action in residual policy learning is a_t and the radial contact force f_r cannot be directly measured. Therefore, we approximate f_r with the following equation

$$f_{r(t)}=f_{t-1}+a_t^{in}\left(f_t-f_{t-1}\right)\Big/ a_t \tag{26}$$

Then the reward $r_t^{in}=\Phi(f_{r(t)}, a_t^{in})$ can be calculated with (23).

With (25), δ and σ are tuned as follows.

$$\delta_t^j=\begin{cases} 0, & \text{if } j=1, t=1 \\ \delta_{T_{j-1}}^{j-1}, & \text{if } j\neq 1, t=1 \\ \delta_{t-1}^j-\kappa\left(\dfrac{e^{\Delta r_t}-e^{-\Delta r_t}}{e^{\Delta r_t}+e^{-\Delta r_t}}\right), & \text{others} \end{cases} \tag{27}$$

$$\sigma_t^j=\begin{cases} \sigma_0, & \text{if } j=1, t=1 \\ \sigma_{T_{j-1}}^{j-1}, & \text{if } j\neq 1, t=1 \\ \xi\dfrac{1-e^{-\Delta r_t}}{1+e^{-\Delta r_t}}\sigma_{t-1}^j, & \text{others} \end{cases} \tag{28}$$

where superscript j is the *j-th* episode, subscript $T_{j\text{-}1}$ is the last time-step of the (j-1)*-th* episode, subscript t is the time-step in one episode, κ and ξ are scale factors.

The action exploration regulation means that the exploration range is decreased with smaller deviation to keep the learning process steady when the agent receives high reward. While the exploration range is increased with larger deviation to increase the learning efficiency when the agent receives small reward.

Moreover, considering the safety in residual policy learning, the saturation function (29) is designed to limit action's exploration range.

$$a^{\mu}=\Gamma\left(a^{\mu}\right)=\begin{cases} a^{\mu}, \text{if } \left|a^{\mu}\right|\leq a_{\mathrm{m}}^{\mu} \\ a_{\mathrm{m}}^{\mu}, \text{otherwise} \end{cases} \tag{29}$$

where a_m^{μ} is the designed threshold to limit action exploration.

4) *Prioritized Experience Replay*

With NAF, residual policy $\Pi_{\mu}(s)$ can be learned from past experience. It is time-consuming and laborious to obtain enough experience in real-world precision insertion. Therefore, it is necessary to improve the efficiency of past experience utilization in residual policy learning. Prioritized experience strategy can make the experience replay more efficient via letting important experience more frequently replayed. Besides, one-step TD error means the distance between the experience's actual action-value and its next-step estimation, which can be used to design experience's replay prioritization. For each experience in experience memory R, we define a probability function as

$$P(i)=\frac{p_i}{\sum_{i=1}^{v} p_i} \tag{30}$$

where the experience can be described as a tuple (s_i, a_i, r_i, s_{i+1}), v is the number of tuples in experience memory R, p_i is the one-step TD error of *i-th* experience.

$$p_i=\left|r_i+\gamma_i Q\left(s_{i+1}, a_{i+1}\right)-Q\left(s_i, a_i\right)\right|+\varepsilon \tag{31}$$

where ε is a small positive constant.

The probability function represents the replay priority of each experience in replay experience memory R. The experience with higher probability can be sampled more frequently in replay process, which makes the learning of insertion policy more efficient. Therefore, the designed prioritized experience replay strategy can accelerate the learning of insertion policy.

The proposed precision insertion policy learning method is summarized in Algorithm 1.

Algorithm 1 Precision insertion policy learning

```
// Calculate initial policy $\Pi_{in}(s)$
1:  Initialize the dataset $D_0$ with demonstrations
2:  Learn $\theta$ via computing optimization function (8)
3:  Compute corresponding policy $\Pi_{in}(s')$ for a new $s'$ with (15)
// Learn residual policy $\Pi_{\mu}(s)$
1:  Initialize number of episodes $M$, the experience replay memory $R$.
    number of steps for each episode $T$, batch size $N$, discount factor $\gamma$,
    delayed update factor $\tau$
2:  For $n$=1,…,$M$ episodes do
3:    Sample initial state $s_0$ from the insertion environment
4:    For $t$=0 to $T$ do
5:        Observe system's state $s_t$
6:        Get execute action $a_t$ with (24) and (15)
7:        Observe next state $s_{t+1}$ and compute reward $r_t$ with (21)
8:        Store transition tuple ($s_t$, $a_t$, $r_t$, $s_{t+1}$) into $R$
9:        Calculate transition's TD error using (31)
10:       Calculate the transition's priority using (30)
11:       Sample $N$ transitions from $R$ based on the priority.
12:       Update parameters of action exploration with (27) and (28)
13:       Update evaluation and target network with (19) and (20)
14:   End for
15: End for
```

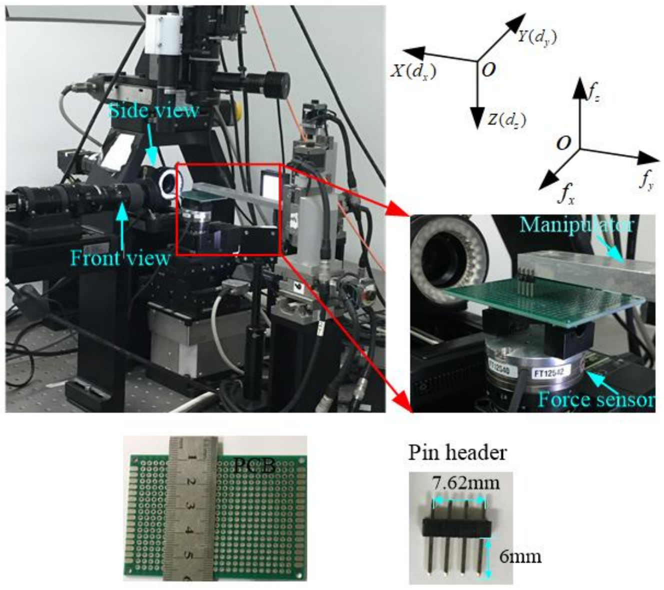


Fig. 4. Experimental platform of precision assembly system.

IV. Experiments

The performance of the proposed insertion policy learning method is comprehensively evaluated on established precision assembly platform. Firstly, the effectiveness of the designed policy mapping model in learning initial policy is verified. Secondly, the learning performance of the residual policy is tested. Thirdly, the insertion policy learned from the proposed method is tested.

A. Experiment Setup

1) Experimental Platform

The experimental system for multiple peg-in-hole precision insertion is established, as shown in Fig. 4. In the system, there are three microscopic cameras including one PointGrey camera and two GC2450 cameras. All cameras are equipped with zoom lens with magnification 0.47~4.5×, whose image sizes are 2448×2050 pixel. The force sensor is ATI Nano-43 with measurement range ±18 N and resolution 1/128N, respectively.

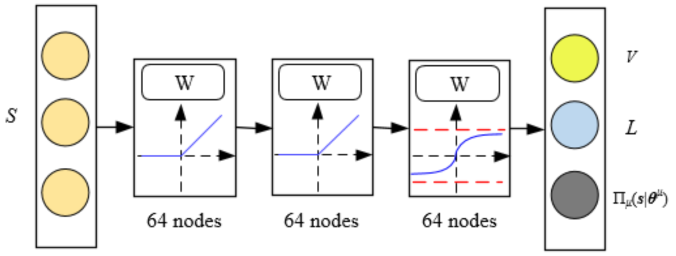


Fig. 5 Network architecture of NAF.

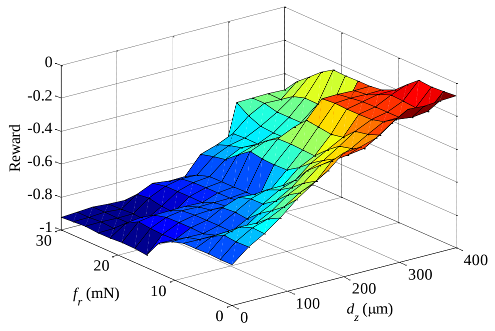


Fig. 6. Fuzzy reward function.

A low pass filter, i.e., Butterworth filter, is designed to filter the noise in the captured force information [5]. The manipulator is a Suguar KWG06030-G, which has three translational degrees of freedom (DOF), and the translational resolution is 1 μm.

The pin header and printed circuit board (PCB) used to verify the effectiveness of the proposed insertion policy learning method are also shown in Fig. 4. The pin headers are commonly seen in MEMS, and they are installed on the PCB forming the electronic device. The pin header with four pins is fixed by the manipulator manually. The force sensor and PCB are fixed on the assembly platform. The diameter of each pin is about 880 μm and diameter of the PCB's aperture is about 900 μm. It seems that the assembly tolerance is about 20 μm. Actually, there are manufacture errors between adjacent pins' distance and holes' distance as well as the in-measurable interference among multiple pins during insertion process, which leads to the slight interference fit between the pin header and PCB. Before insertion, the pose alignment between the pin header and PCB is achieved based on image-based visual servo [5]. In insertion process, the force sensor is used to measure the contact forces, and manipulator is an actuator that is used to adjust pin header's position.

2) Network Architecture of NAF

Considering the nonlinear dynamics of precision insertion, the architecture of the designed evaluation network is shown in Fig. 5. The input of the evaluation network is state s, and the output consists of the value function V, the matrix L, and action $\Pi_\mu(s|\theta^\mu)$. The two hidden layers with 64 Relu units are designed to avoid the "gradient vanish", and the output layer is a tanh layer to limit the output.

The discount factor is set γ=0.98 and the learning rate of the Adam optimizer is set as 0.001. The soft update rate of the target network is set as τ=0.001. Furthermore, the size of the mini-batch is 64 and the size of the experience memory R is 10^6. The initial deviation is set as μ_0=10, and the scale factors are set as κ=0.8 and ξ=0.95.

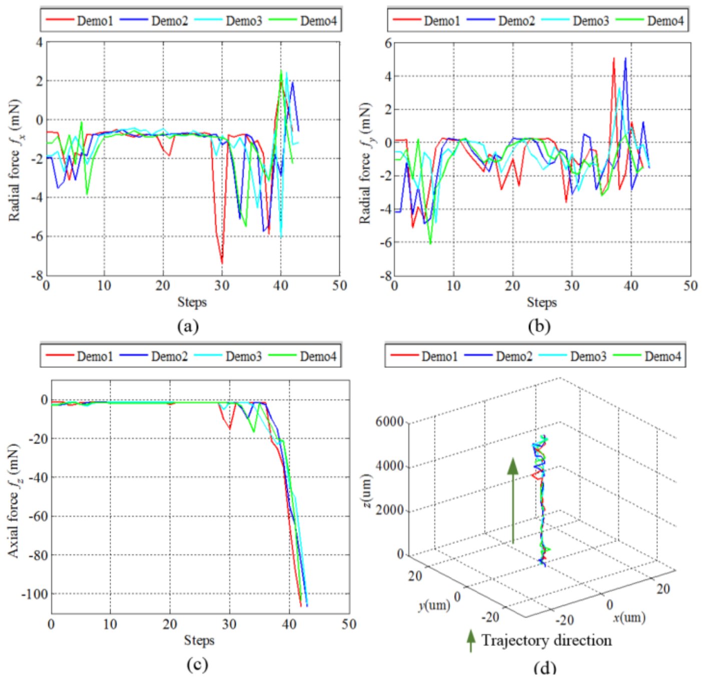


Fig. 7. Results of four insertion demonstrations. (a)-(c) Contact forces f_x, f_y and f_z. (d) Movement trajectories of the manipulator.

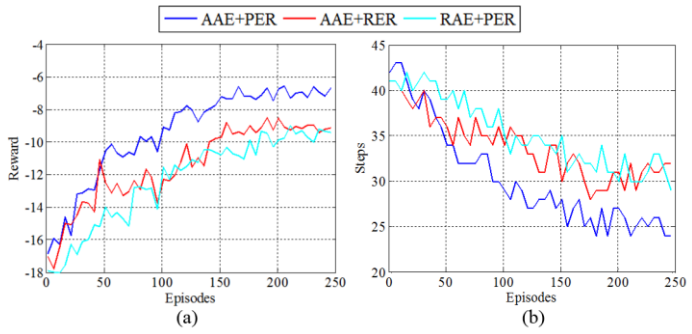


Fig. 8. Performance of residual policy learning. (a) Reward. (b) Steps.

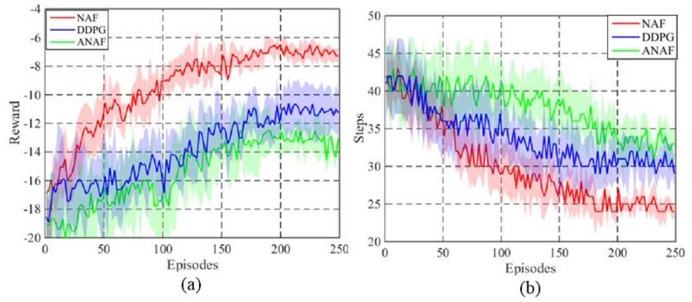


Fig. 9. Learning curves for residual policy with NAF, DDPG and ANAF. (a) Reward. (b) Steps.

3) Fuzzy Reward Function

The inputs of the fuzzy reward function are the radial contact force f_r and the axial feeding step depth d_z, and the output is the reward r_2, as shown in Fig. 6. The range of reward r_2 is limited within [-1, 0], the axial feeding step length d_z is designed within [0, 400] μm, and the radial contact force f_r used to protect the pin header is set within [0, 30] mN. The maximum number of insertion steps $k_{\max}$ is 50. Once the captured radial contact force f_r is out of the range or the number of insertion steps exceeds the maximum $k_{\max}$, the insertion experiment is terminated and the reward is r_2=-1.

B. Insertion Policy Learning

1) Initial Policy Learning

Firstly, four demonstrations are carried out manually, and the demonstration data are stored in dataset D_0. In specific, the manipulator is controlled to move down the pin header, meanwhile modulating its lateral position to maintain the radial contact forces in acceptable range. After each action, the state s consisting of contact forces and the action a including the translational movements along axes X, Y and Z are stored in dataset D_0 as a tuple (s, a) in demonstrations. Then the designed policy mapping model is learned based on the collected demonstration data in dataset D_0. For given state s, the initial policy can be determined from GMR.

The contact forces and the movement trajectories of the four insertion demonstration experiments are shown in Fig. 7. In specific, Fig. 7 (a)-(c) show the contact forces f_x, f_y and f_z in demonstrations. Fig. 7(d) shows the manipulator's movement trajectories in demonstrations. The movement along Z axis represents the axial feeding length to complete precision insertion, and the movements along X and Y axes represent the manipulator's radial adjustments to maintain the radial contact forces in small range.

2) Residual Policy Learning

To test the learning performance of the proposed adaptive action exploration and prioritized experience replay strategies, comparative experiments with different action exploration and experience replay strategies are carried out. Fig. 8 shows the learning performance with different action exploration and experience replay strategies: adaptive action exploration and prioritized experience replay (AAE+PER), adaptive action exploration and random experience replay (AAE+RER), random action exploration, i.e., Gaussian noise with constant parameters and prioritized experience replay (RAE+PER). In specific, Fig. 8(a) shows the change of accumulated reward in residual policy learning, and Fig. 8(b) shows the change of the number of insertion steps in residual policy learning.

From Fig. 8, it can be seen that the learning speed with AAE+PER is increased compared to the learning speed with other comparative methods. Specifically speaking, the accumulated reward with AAE+PER can achieve a higher value after 150 episodes' learning. Meanwhile, the number of insertion steps in each episode is effectively decreased, which acts in accordance with the purpose of the designed precision insertion policy learning method. Moreover, the learning curve with AAE+PER is smoother than the learning curves with other comparative methods, which is helpful for insertion policy learning in real-world environment.

The reasons for us choosing the residual policy learning with AAE+PER are as follows. The adaptive action exploration strategy is updated after each action according to the variation of reward increment, which is helpful to increase the efficiency of learning residual policy. Furthermore, different from random consistent experience replay method which samples experience with the same probability, the prioritized experience replay method assigns a higher priority to the experience with larger

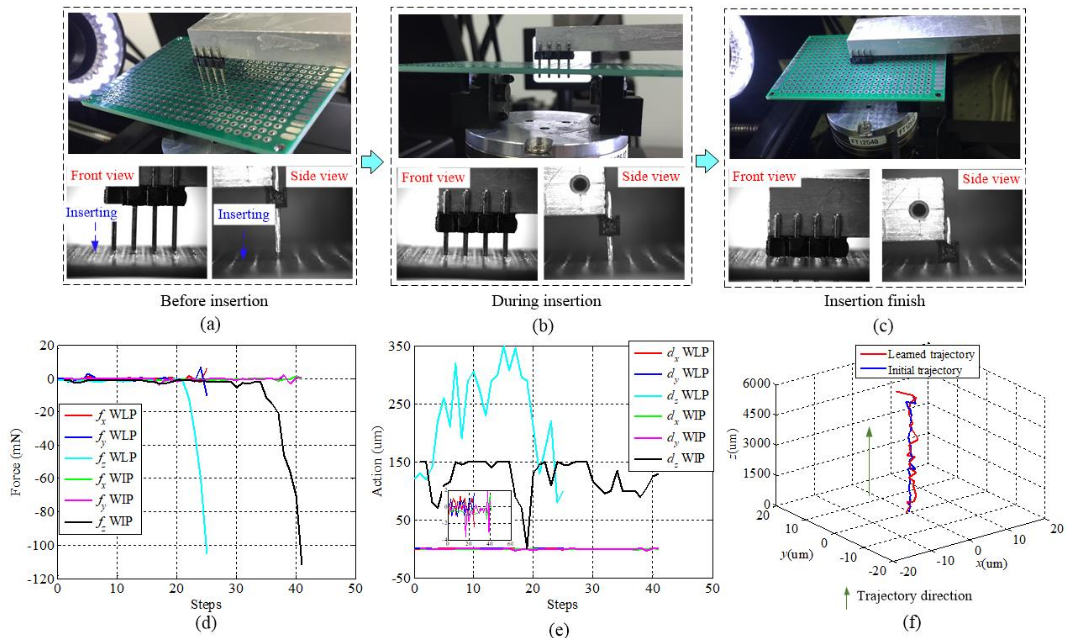


Fig. 10. Results of pin header insertion experiment with learned proposed method. (a) Images before insertion. (b) Images during insertion. (c) Images after insertion. (d) Contact forces f_x, f_y and f_z. (e) Actions d_x, d_y and d_z. (f) Movement trajectory of the manipulator. (WLP--with learned policy, WIP--with initial policy).

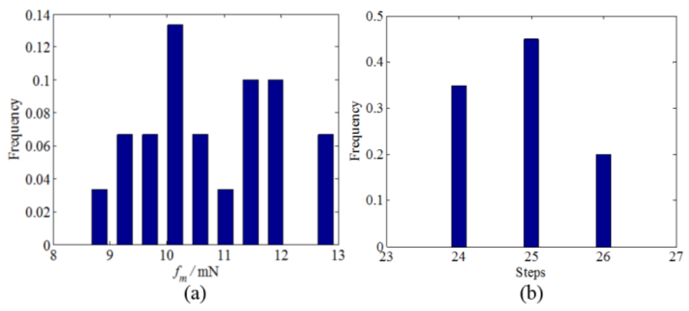


Fig. 11. Distributions of the maximum of radial force f_m and the number of steps of 20 times experiments. (a) Maximum of radial force f_m. (b) Number of steps.

TD error since larger TD error means the experience is more useful in insertion policy learning. Therefore, combining the adaptive action exploration and prioritized experience replay strategies can effectively improve residual policy learning performance.

3) *Learning Performance Comparative Evaluation*

The learning performance comparative experiments are conducted to verify the advantage of the proposed NAF method. We choose DDPG as the comparative method, which is a state of art algorithm and is regarded as one of efficient off-policy reinforcement learning methods. Besides, the ablation experiment of learning residual policy without initial policy's acceleration (ANAF) is also conducted.

For each algorithm, we train five different instances with different random seeds. The learning curves are presented in Fig. 9. The solid curve represents the mean of five trials, and the shaded region represents the standard deviation of five trials. The reward is considered to convergence if its variation is less than 1. It means the stable insertion policy is learned. From Fig. 9, it can be seen that the proposed NAF method outperforms comparative algorithms in both final performance and learning efficiency. In specific, NAF requires about 160 episodes to learn a stable insertion policy. DDPG requires about 200 episodes to learn a stable insertion policy. ANAF requires about 250 episodes to learn a stable insertion policy.

NAF yields better performance than DDPG because the adaptive action exploration and prioritized experience replay strategies are well designed in NAF, and NAF trains less network hyper-parameters than DDPG. In addition, because the agent directly explores in complex insertion environment without any prior knowledge in ANAF, the asymptotic performance of ANAF is worse than NAF. And it needs more trials to learn an optimal policy with ANAF.

C. *Precision Insertion Experiments*

1) *Insertion Experiments*

To verify the advantage of the learned policy, the insertion experiments are conducted. The experimental results are shown in Fig. 10. Fig. 10(a)-(c) show the captured images by two side microscopic cameras and human. It can be seen that the precision insertion can be well accomplished with the learned policy. Furthermore, Fig. 10(d)-(f) show contact forces, actions and manipulator's movement trajectory in the insertion process with the learned policy and the initial policy. From Fig. 10(d)-(f), it can be seen that the insertion assembly can be achieved within 25 steps with the learned policy. While it requires 41 steps to achieve insertion assembly with the initial policy. Therefore, the insertion efficiency of the learned policy is higher than the efficiency of the initial policy from the demonstrations. Besides, when the pin header is moved down in insertion process, the radial adjustments along X and Y axes are regulated to reduce the radial contact forces with both policies, but the manipulator's radial adjustments are more flexible with the learned policy, which makes the insertion process more compliant.

Besides, repetitive insertion experiments are conducted to test the learned insertion policy. The maximum radial contact force f_m=max{f_r} and the number of insertion steps are shown

TABLE I
ADAPTABILITY TO INITIAL POSITION ERRORS

NO.	Position errors (μm)	Steps	f_m (mN)	NO.	Position errors (μm)	Steps	f_m (mN)
1	(20, 0)	27	16.2	6	(0, -15)	26	11.7
2	(0, 20)	27	18.2	7	(-9, 10)	26	14.1
3	(8, -5)	24	15.4	8	(5, -10)	25	16.3
4	(10, -5)	25	15.9	9	(-9, -9)	24	14.2
5	(-10, 5)	26	13.1	10	(5, -5)	24	15.7

TABLE II
COMPARISON OF LEARNED INSERTION POLICY WITH PROPOSED METHOD AND METHOD [13]

Proposed method			Method [13]		
NO.	Steps	f_m (mN)	NO.	Steps	f_m (mN)
1	24	12.43	1	32	15.63
2	24	13.35	2	29	18.89
3	26	11.09	3	30	18.45
4	26	12.10	4	31	17.01
AVE	25	12.24	AVE	31	19.50
SD	1.15	0.93	SD	1.29	1.48

in Fig. 11. From Fig. 11, it can be seen that the radial contact force is maintained within 13 mN, and the number of insertion steps is less than 26. The experimental results verify the repeatability of the learned insertion policy, which is helpful to apply the proposed insertion policy learning method to other repetitive insertion tasks.

2) *Adaptability Experiments*

To be more persuasive, the adaptability of the proposed insertion policy learning method to initial position errors (Δx, Δy) is verified. Ten insertion experiments with different initial position errors are carried out. The number of insertion steps and the maximum radial contact force f_m of ten insertion experiments are listed in Tab. I. From Tab. I, it can be seen that the proposed method has adaptability to initial position errors. Particularly, the position errors lead to the variation of contact dynamics in precision insertion, and the learned insertion policy needs to overcome position errors. Thus, the number of insertion steps and radial contact force f_m change as the position errors' variation. The results show that the proposed insertion policy learning method has robustness to initial position errors.

3) *Generalization Experiments*

Firstly, the insertion experiments are conducted with the proposed method and comparative method [13] for the same kind of pin header component. Comparative method [13] is a classic insertion policy learning method. Totally four insertion experiments for the proposed method and comparative method [13] are conducted. The results are shown in Tab. II. The results consist of the number of insertion steps, the maximal radial contact force, the average (AVE) and the standard deviation (SD) with the learned insertion policies. From Tab. II, it can be seen that the insertion task can be finished in 24 to 26 steps with the proposed method, whose efficiency is higher than that with comparative method [13]. Besides, with the proposed method, the maximal radial contact forces are less than 13.35 mN with SD 0.93 mN, which are smaller than that with comparative method [13]. It demonstrates that the learned insertion policy with the proposed method has better performance than the learned insertion policy with comparative method [13] both in efficiency and uniformity.

To further validate the generalization of proposed method, the precision insertion experiments of a triode into the PCB are conducted with the proposed method and comparative method [13]. The triode has 3 pins, and the distance between triode's adjacent pins is 2.54 mm. The fitting type between triode and PCB is interference fit, and the insertion depth is about 1 cm.

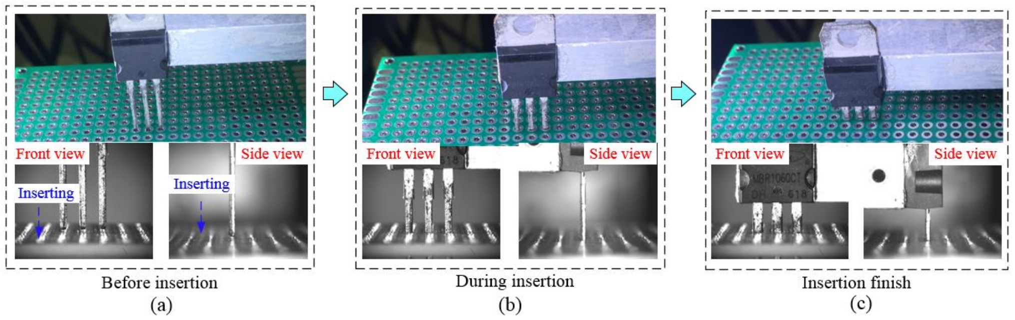
The precision insertion experimental results with the proposed method are shown in Fig. 12. Fig. 12 (a)-(c) show the captured images by two side microscopic cameras and human. From Fig. 12 (a)-(c), it can be seen that the triode can be steadily and efficiently inserted into the PCB without damage. Besides, the contact forces, the actions and the manipulator's movement trajectory in insertion process are shown in Fig. 12 (d)-(f). It can be seen that the insertion experiment can be completed within 42 steps. Besides, via manipulator's radial adjustments, the radial contact forces are well controlled within safe range. These results demonstrate that the insertion policy learned from the proposed method tends to be safer and more efficient, which coincides with the goal of precision insertion.

The experimental results with comparative method [13] are shown in Fig. 13. From the results, it can be seen that there are large radial contact forces, and the radial contact force exceeds the maximum contact force threshold in the last few steps of insertion experiments. It means that the comparative method [13] cannot successfully realize the triode insertion task.

It can be seen that the proposed insertion policy learning method can also be used to other multiple peg-in-hole insertion tasks. However, it should be noted that the insertion policy needs to be relearned once the insertion environment changes due to the contact characteristics for different insertion tasks are various. In specific, the insertion environment mainly consists of intrinsic and external insertion environment. If the kind of multi-peg component, the number or the size of pegs is changed, the intrinsic insertion environment is changed. Besides, if the hardware or installation of the assembly platform is changed, the external insertion environment is changed. Furthermore, the designed hierarchical reward function is available to the situations that need to balance multiple factors, such as efficiency, safety and others. It is not necessary for single pin-hole clearance insertion to use hierarchical reward because of less factors of task. The adaptive action exploration strategy is always helpful for accelerating the learning. If the historical experiences have different effect in learning process, the prioritized experience replay strategy is very useful. Otherwise, the prioritized experience replay strategy is not necessary if all historical experiences are perfect.

V. CONCLUSIONS

In this paper, an efficient insertion policy learning method for multiple peg-in-hole precision assembly is proposed. Firstly, the insertion policy is separated into initial policy and residual policy. The initial policy is learned from policy mapping model. In specific, the policy mapping model is based on GMM, and the initial policy can be learned by generalizing obtained policy mapping model. It takes advantage of the human insertion experience and effectively accelerates initial policy's learning.



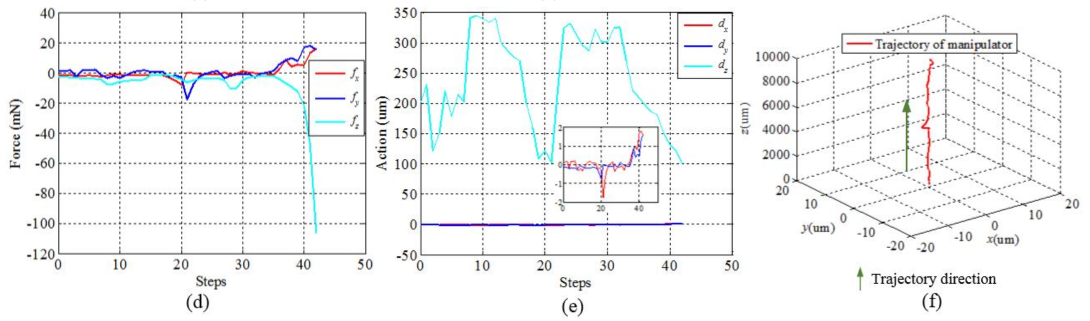


Fig. 12. Results of triode precision insertion experiment with proposed method. (a) Images before insertion. (b) Images during insertion. (c) Images after insertion. (d) Contact forces f_x, f_y and f_z. (e) Actions d_x, d_y and d_z. (f) Movement trajectory of the manipulator.

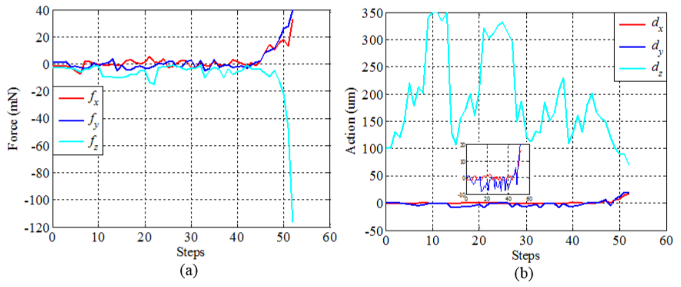


Fig. 13. Results of triode precision insertion experiment with method [13]. (a) Contact forces f_x, f_y and f_z. (b) Actions d_x, d_y and d_z.

Besides, the residual policy is well learned based on modified NAF. The adaptive action exploration strategy and prioritized experience replay mechanism are combined to improve the learning speed of the residual policy. Finally, comprehensive precision insertion experiments are conducted on established precision assembly platform to validate the effectiveness of the proposed insertion policy learning method.

The proposed method is for assembly of rigid components, which is not applicable to flexible components. It also requires human design the reward function in reinforcement learning. In the future work, we will investigate how to assemble flexible components with skill learning methods. Besides, inverse reinforcement learning will be combined to reinforcement learning to improve the accuracy of reward function.

References

[1] S. Liu, D. Xing, Y. Li, J. Zhang, and D. Xu, "Robust insertion control for precision assembly with passive compliance combining vision and force Information", *IEEE/ASME Trans. Mechatron.*, vol. 24, no. 5, pp. 1974-1985, Oct. 2019.

[2] D. Xing, D. Xu, F. Liu, H. Li and Z. Zhang, "Precision assembly among multiple thin objects with various fit types", *IEEE/ASME Trans. Mechatron.*, vol. 21, no. 1, pp. 364-378, Feb. 2016.

[3] F. Shen, W. Wu, D. Yu, D. Xu and Z. Cao, "High precision automated 3-D assembly with attitude adjustment performed by LMTI and vision-based Control", *IEEE/ASME Trans. Mechatron.*, vol. 20, no. 4, pp. 1777-1789, Aug. 2015.

[4] D. Xing, F. Liu, S. Liu, and D. Xu, "Efficient insertion of partially flexible objects in precision assembly", *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 706-715, Apr. 2019.

[5] S. Liu, D. Xu, D. Zhang, and Z. Zhang, "High precision automatic assembly based on microscopic vision and force information", *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 1, pp. 382-393, Jan. 2016.

[6] Q. Xu, "Robust impedance control of a compliant microgripper for high speed position/force regulation", *IEEE Trans. Ind. Electron.*, vol. 62, no. 3, pp. 1201-1209, Feb. 2015.

[7] S. Liu, Y. Li, D. Xing, and D. Xu, "An efficient insertion control method for precision assembly of cylindrical components", *IEEE Trans. Ind. Electron.*, vol. 64, no. 12, pp. 9355-9365, Dec. 2017.

[8] Z. Hou, H. Dong, K. Zhang, Q. Gao, K. Chen and J. Xu, "Knowledge-driven deep deterministic policy gradient for robotic multiple peg-in-hole assembly tasks", in *Proc. IEEE Int. Conf. Robot. Biomimetics*. Kuala Lumpur, Malaysia, Dec. 12-15, 2018, pp. 256-261.

[9] C. Yang, C. Zeng, C. Fang, W. He and Z. Li, "A DMPs-based framework for robot learning and generalization of humanlike variable impedance skills," *IEEE/ASME Trans. Mechatron.*, vol. 23, no. 3, pp. 1193-1203, Jun. 2018.

[10] C. Yang, C. Zeng, Y. Cong, and N. Wang, "A learning framework of adaptive manipulative skills from human to robot," *IEEE Trans. Ind. Inform.*, vol. 15, no. 2, pp. 1153-1161, Feb. 2019.

[11] T. Inoue, G. D. Magistris, A. Munawar, T. Yokoya, and R. Tachibana, "Deep reinforcement learning for high precision assembly tasks", in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Vancouver, BC, Canada, Sep. 24-28, 2017, pp. 819-825.

[12] J. Luo, E. Solowjow, C. Wen, J. A. Ojea and A. M. Agogino, "Deep reinforcement learning for robotic assembly of mixed deformable and rigid objects", in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Madrid, Spain, Oct. 1-5, 2018, pp. 2062-2069.

[13] J. Xu, Z. Hou, W. Wang, B. Xu, K. Zhang, and K. Chen, "Feedback deep deterministic policy gradient with fuzzy reward for robotic multiple peg-in-hole assembly tasks", *IEEE Trans. Ind. Inform.*, vol. 15, no. 3, pp. 1658-1667, Mar. 2019.

[14] G. Wen, C. L. Chen, and S. Ge, "Optimized adaptive nonlinear tracking control using actor–critic reinforcement learning strategy," *IEEE Trans. Ind. Inform.*, vol. 15, no.9, pp. 4969-4977, Sep. 2019.

[15] G. Schoettler, A. Nair, J. Luo, S. Bahl, J. A. Ojea, E. Solowjow, and S. Levine, "Deep reinforcement learning for industrial insertion tasks with visual inputs and natural rewards," arXiv preprint arXiv: 1906.05841.

[16] F. Qin, D. Xu, D. Zhang, and Y. Li, "Robotic skill learning for precision assembly with microscopic vision and force feedback," *IEEE/ASME*

Trans. Mechatron., vol. 24, no. 3, pp.1117-1128, Jun. 2019.

[17] Y. Wang, R. Xiong, H. Yu, J. Zhang, and Y. Liu, "Perception of demonstration for automatic programing of robotic assembly: framework, algorithm, and validation", *IEEE/ASME Trans. Mechatron.*, vol. 23, no. 3, pp. 1059-1070, Jun. 2018.

[18] D. Ehlers, M. Suomalainen, J. Lundell, and V. Kyrki, "Imitating human search strategies for assembly", in *Proc. IEEE Int. Conf. Robot. Autom.*, Montreal, Canada, May 20-24, 2019, pp. 7821-7827.

[19] S. Christen, S. Stevsic, and O. Hilliges, "Demonstration-guided deep reinforcement learning of control policies for dexterous human-robot interaction", in *Proc. IEEE Int. Conf. Robot. Autom.*, Montreal, Canada, May 20-24, 2019, pp. 2161-2167.

[20] T. Hester, M. Vecerik, O. Pietquin, and M. Lanctot, "Deep Q-learning from demonstrations", arXiv:1704.03732v4 [cs.AI] 22 Nov 2017.

[21] G. V. Cruz, Y. Du, and M. E. Taylor, "Pre-training neural networks with human demonstrations for deep reinforcement learning", arXiv preprint arXiv:1709.04083.

[22] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations", arXiv preprint arXiv: 1709.10087.

[23] Z. Deng, H. Guan, R. Huang, H. Liang, L. Zhang, and J. Zhang, "Combining model-based Q-Learning with structural knowledge transfer for robot skill learning", *IEEE Trans. Cogn. Dev. Syst.*, vol. 11, no. 1, pp. 26-35, Mar. 2019.

[24] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyl, J. A. Ojea, E. Solowjow, and S. Levine, "Residual reinforcement learning for robot control", in *Proc. IEEE Int. Conf. Robot. Autom.*, Montreal, Canada, May 20-24, 2019, pp. 6023-6029.

Yanqin Ma received the B.Sc. degree in automation from Qingdao College of Ocean University of China in 2012, received the M.Sc. degree from Harbin Engineering University in 2015, and received the Ph.D. degree in control science and engineering from Institute of Automation, Chinese Academy of Sciences (IACAS) and University of Chinese Academy of Sciences (UCAS), Beijing, China in 2020.

She is currently a Lecturer at College of Automation, Nanjing Institute of Technology, Nanjing, China. Her current research interests include visual measurement, robot control, micro-assembly and machine learning.

De Xu (M'05–SM'09) received the B.S. and M.S. degrees from the Shandong University of Technology, Jinan, China, in 1985 and 1990, respectively, and the Ph.D. degree from the Zhejiang University, Hangzhou, China, in 2001, all in control science and engineering.

Since 2001, he has been with the Institute of Automation, Chinese Academy of Sciences, Beijing, China, where he is currently a Professor with the Research Center of Precision Sensing and Control. His current research interests include robotics and automation, such as visual measurement and control, microscopic vision, micro-assembly, and skill learning.

Fangbo Qin received the B.Sc. degree in automation from Beijing Jiaotong University, Beijing, China, in 2013, and the Ph.D. degree in control science and engineering from Institute of Automation, Chinese Academy of Sciences (IACAS) and University of Chinese Academy of Sciences (UCAS), Beijing, China.

He is currently an Assistant Professor at IACAS. His research interests include robot vision, deep learning, and intelligent robotic manipulation.