https://doi.org/10.26599/AIR.2024.9150043 CAAI Artificial Intelligence Research

Reinforcement Learning-Based Sequential Control Policy for Multiple Peg-in-Hole Assembly

Xinyu Liu[1], Chao Zeng[2] ✉, Chenguang Yang[2], and Jianwei Zhang[3]

ABSTRACT

Robotic assembly is widely utilized in large-scale manufacturing due to its high production efficiency, and the peg-in-hole assembly is a typical operation. While single peg-in-hole tasks have achieved great performance through reinforcement learning (RL) methods, multiple peg-in-hole assembly remains challenging due to complex geometry and physical constraints. To address this, we introduce a control policy workflow for multiple peg-in-hole assembly, dividing the task into three primitive sub-tasks: picking, alignment, and insertion to modularize the long-term task and improve sample efficiency. Sequential control policy (SeqPolicy), containing three control policies, is used to implement all the sub-tasks step-by-step. This approach introduces human knowledge to manage intermediate states, such as lifting height and aligning direction, thereby enabling flexible deployment across various scenarios. SeqPolicy demonstrated higher training efficiency with faster convergence and a higher success rate compared to the single control policy. Its adaptability is confirmed through generalization experiments involving objects with varying geometries. Recognizing the importance of object pose for control policies, a low-cost and adaptable method using visual representation containing objects' pose information from RGB images is proposed to estimate objects' pose in robot base frame directly in working scenarios. The representation is extracted by a Siamese-CNN network trained with self-supervised contrastive learning. Utilizing it, the alignment sub-task is successfully executed. These experiments validate the solution's reusability and adaptability in multiple peg-in-hole scenarios.

KEYWORDS

multiple peg-in-hole assembly; deep reinforcement learning; self-supervised contrastive learning

With the development of robotics and artificial intelligence (AI) techniques, Industry 4.0 is thriving and revolutionizing manufacturing processes through automation and digitization. Robots have been employed across various industry scenarios to automate mass manufacturing and enhance production efficiency, with assembly being one of the most critical and everyday industrial operations[1, 2]. Consequently, robotic assembly, particularly peg-in-hole assembly, has been extensively researched for many years. The traditional strategy involves pre-defining the assembly process and pre-programming corresponding operations. While this method can achieve high assembly precision, the system's deployment is complex and challenging to adapt to various situations, like changing grasping points and diverse objects' geometry. Then reinforcement learning (RL) methods, which have achieved distinguishing performance in robotic control, are applied to train control policies for the peg-in-hole task, achieving high precision and robustness against environmental uncertainties[3–5]. Several studies have demonstrated high performance in single peg-in-hole assembly using RL-trained control policies[3, 6, 7]. Compared with single peg-in-hole assembly, the multiple peg-in-hole assembly has broader application scenarios, like vehicle hub assembly and electronics assembly, as shown in Fig. 1.

However, the multiple peg-in-hole assembly remains challenging because of the strict successful assembly conditions, complicated geometry and physical interaction. The multiple peg-in-hole assembly follows the same process as a single peg-in-hole assembly, but it is much more challenging with several pegs, as all the pegs should be inserted into corresponding holes simultaneously. Unlike the single peg task, which typically considers 5 or 6 dimensions for rotation and position, the multiple peg assembly requires consideration of 18 dimensions in this study. Additionally, the single peg-in-hole assembly involves a single peg with different cross-sectional shapes (e.g., circular or triangular), but multiple peg-in-hole assembly often include multiple pegs with a handle, like the plugs and sockets. Therefore, the peg's pose is not directly connected with the gripper's pose because the grasping location varies in the handle, and all the pegs' interactions with the hole are different.

There are some successful solutions explored to solve the multiple peg-in-hole assembly task. Researchers typically pre-build the physical interaction model of the peg-in-hole process and train one control policy to implement the task of assembling the peg into the hole based on the model and force signal[8, 9]. However, the solution becomes complex when transferring to various object geometry, and the uncertainty of the peg's in-hand pose is not considered in the constructed simulation environment, which is crucial for the generalization ability of the solution. More recent work[5] constructs one comprehensive simulation environment with domain randomization where one control policy is trained with multimodal observation. The trained control policy has a strong generalization ability to various object geometry and

1 Department of Electrical and Photonics Engineering, Technical University of Denmark, Kongens Lyngby 2800, Denmark
2 Department of Computer Science, University of Liverpool, Liverpool L69 3BX, UK
3 TAMS Group, Department of Informatics, University of Hamburg, Hamburg 22527, Germany
Address correspondence to Chao Zeng, chaozeng@ieee.org

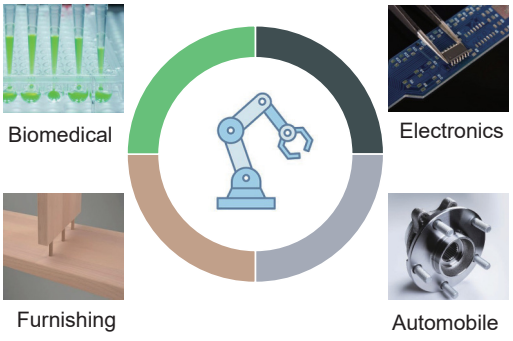

Fig. 1 Application scenarios of multiple peg-in-hole assembly, including biomedical, automobile, furnishing, and electronics industries.

dynamic environments. Nevertheless, the policy learning process in this setup is less effective with complex reward functions and redundant observations over 100 dimensions. The inference speed is slow with output action at 10 Hz, which makes the assembly time to be long over fifteen seconds.

In this study, a flexible and reusable framework for multiple peg-in-hole assembly is proposed to improve the policy learning efficiency and decrease the computational complexity while keeping the control policies' generalization ability. The entire pipeline of the framework is shown in Fig. 2. Additionally, visual representation mainly contains information about objects' poses, which is learned from the RGB images for object pose estimation to further improve the solution's adaptability and computational efficiency. To implement the pipeline, a simulation environment is construed in Isaac Gym with domain randomization to shorten the gap with the real world. Then, the policy learning is conducted in this simulation environment using proximal policy optimization (PPO) algorithms. Instead of training a control policy for the entire task, the sequential control policy (SeqPolicy) is applied to implement these sub-tasks in sequence and finally complete the whole task. Specifically, the task is divided into three common sub-tasks: picking, alignment, and insertion based on human primitive skills. Then, three control policies are trained to master each primitive skill and implement the overall task, and they are independent to modularized pipeline. In the process, they are pre-loaded and transferred based on the peg states.

The impedance controller is introduced to improve system compliance and ensure safety in case of outlier output from the neural network in the control policy. Two experiments demonstrate that the SeqPolicy can achieve the assembly task effectively and adapt to various scenarios. After that, a Siamese-CNN is trained using self-supervised contrastive learning methods based on the constructed dataset to extract the desired visual representation, which is suitable for object pose estimation. With the estimated objects' pose from these visual representations, the previously trained policy can implement the alignment sub-task successfully. The primary contributions of this paper are:

(1) A flexible and reusable framework for the multiple peg-in-hole assembly task. The proposed method, sequential control policy, improves the policy learning efficiency significantly, and the trained control policy can be transferred to similar peg-in-hole scenarios.

(2) The ablation experiment explores compact observation in the multiple peg-in-hole assembly task to decrease computational complexity. Three modalities, pose, force signal, and proprioception, are crucial in this assembly task, and various sensor data in these modalities are combined as observation and tested in policy learning to explore compact and effective observation.

(3) A pose estimation strategy based on the self-supervised contrastive learning method is proposed to provide pose information in the robot base frame directly. The network is small and has a higher inference speed.

The rest of the paper is organized as follows. The background and related work are reviewed in section 2. Then, the technical problem in this study and the proposed methodologies are introduced in sections 3 and 4. Finally, the experiments and results are shown in section 5, followed by a conclusion and discussion in section 6.

1 Related Work

1.1 Deep reinforcement learning-based robotic assembly

Robotic assembly, a popular topic related to manufacturing, has been studied for decades. Traditionally, manual programming is used to design robots' motion plans given specific tasks, which is reliable but sensitive to uncertain parameters, especially in

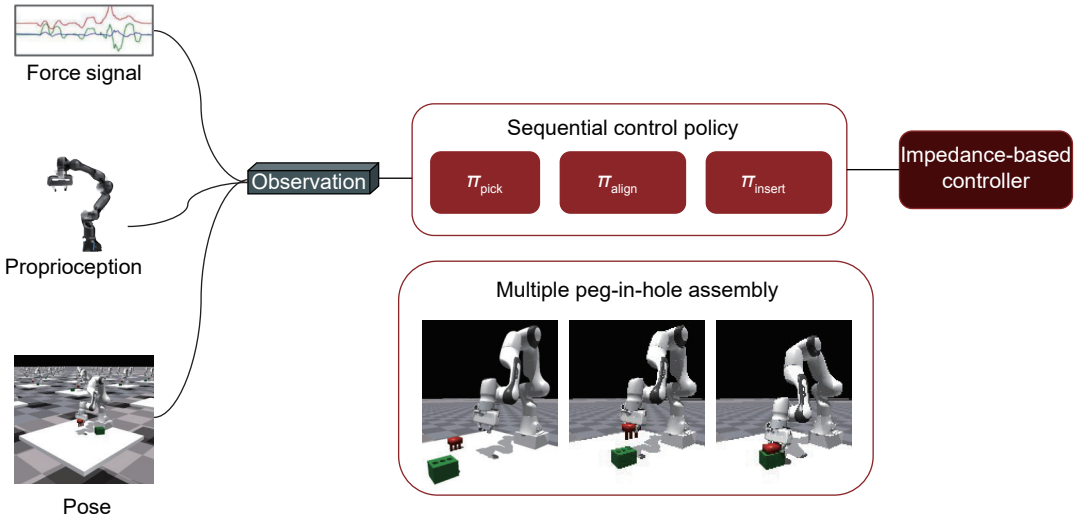

Fig. 2 Overview of proposed solution. In the simulation environment, the force signal and proprioception data are collected from the specific sensors, and the pose information is provided by visual representation. Then, these data are combined as the observation for the sequential control policy. The Cartesian action output from the control policy is then transferred to the joint actuation in joint space to control the end-effector move to the desired position.

unstructured environments[10, 11]. Recently, deep reinforcement learning (DRL) has been extensively applied in robotic control[10, 12]. DRL incorporates neural networks into the control policy for state and value approximation, enabling the control policy to process high-dimension action and observation space, which is essential for robotic control tasks[13, 14].

Several research pre-defines the physical model of the entire peg-in-hole process, using the force signal as observations for control policy to estimate the specific states in the process, and the trained control policy generates the trajectory to guide the robot to implement the whole task with a high precision[8, 9, 15, 16]. These control policies also have the adaptability to specific variables and unforeseen environmental situations[17, 18]. For example, the control policy trained by the DDPG algorithm can implement a peg-in-hole assembly task with the uncertain hole's position[3]. Additionally, several works implement the peg-in-hole assembly task efficiently and precisely by integrating human knowledge or human demonstration[19–21]. In similar peg-in-hole tasks, the researchers use the Ape-X DDPG algorithm to train the control policy based on human demonstrations to assemble the timber part into the mating element[22]. The trained control policy could be transferred to other peg-in-hole tasks, including nut-bolt-place and LAN port insertion. To further improve the generalization of the control policy, researchers[5, 7, 8] introduce the visual modality into the observation space. Compared to the proprioceptive modalities, visual representation containing explicit semantic information is beneficial to the task. The trained control policies can implement the peg-in-hole assembly in various scenarios. According to previous robotic control strategies, one approach sets the joints' actuation as actions. Therefore, the control policy outputs torque/force directly to control the robot. The other approach combines the controller and control policy, where the control policy will output action in Cartesian space and then transfer to joint space using the controller. Considering the black-box property of the neural network, this study combines the Cartesian space impedance controller with the sequential control policy to ensure safety in the process.

1.2 Long-horizon planning task

In the practical application fields, there are many long-horizon tasks across different intermediate states, like dexterous hand manipulation[23, 24]. Training a control policy that could guide robots through these tasks is challenging due to errors accumulating throughout the process and the complicated success conditions. Moreover, the end states of some key intermediate states are crucial for overall success. For example, in the peg-in-hole assembly, the end state of the picking step is important for the next step: alignment. If the peg is not correctly oriented, the control policy may repeatedly attempt to adjust it or get stuck in the suboptimal robot gestures, eventually leading to failure.

Dividing the long-horizon task into reusable and basic sub-tasks, each implemented with specific control policies in sequence, is an effective and proven method[4, 25]. The sub-task is typically simple and primitive, making the training setup explicit. Specifically, the reward function is clear and straightforward. Additionally, the sub-task can also be divided by the interaction type, such as continuous contact and discontinuous contact[19] stage, which are suitable for model-based and model-free methods separately. Several studies successfully use the control policies for a series of sub-tasks to implement the long-horizon task[26, 27]. The policy training methods for sub-tasks include policy learning from demonstration[28–30], unsupervised exploration[31], and manual-defined methods[32, 33]. The sub-tasks in this study are the classical robotic manipulation task[34], so the PPO algorithm[35] is used to train control policies for them.

Policy chaining is important for sequential control policies, as it ensures a valid high-level trajectory plan and seamless transition among sub-tasks. Adding the terminate states of the preceding policy[36, 37] is helpful in estimating the transition state. However, compression of terminate states is challenging in complex tasks like humanoid operations with high-dimensional action or state space. Thus, training a discriminator to estimate the terminate states is an effective solution[38]. Hierarchical reinforcement learning can also be used to decide the stages of complex tasks while integrating human knowledge[21]. In the peg-in-hole assembly task, the state of the peg can directly indicate the states, lifted, aligned, or inserted. Therefore, in this study, the peg's state is directly used to identify these different intermediates. The corresponding control policy is applied at each intermediate and will be transferred to the following policy once the peg reaches the next intermediate steadily.

1.3 Representation learning for robotic manipulation

Representation Learning is a helpful technique for extracting meaningful patterns from raw data to create representations that encode the significant information or invariant features behind the data[39]. In robotic manipulation, this technique can be used to extract low-dimensional representation containing semantic information from high-dimensional data, especially for capturing dynamic or geometric information from images[13]. The learnt low-dimensional representation could be used directly in the policy learning process for the specific manipulation task. References [5, 7] encode the multimodal representations from four different sensors, which are then used for policy learning. The trained control policy can implement the single peg-in-hole task and be transferred to different object geometries. In the research of multiple peg-in-hole tasks, the visual representation, including key features in RGB images, is used as part of observation in policy learning. The control policy can implement the multiple peg-in-hole task and has a strong generalization ability to various cross-sectional shapes.

Except for the direct involvement in the policy learning, representations with semantic information can also be used to provide pose information explicitly and flexibly instead of manual calibration each time in new scenarios. References [40, 41] use the contrastive learning method to learn invariant and transformative representations for the 3D pose estimation of hand and household objects. Because of the effectiveness of the large model, some works[42, 43] apply the CLIP[44] and visual-language model to understand the scenarios and provide pose information for the grasping or placing task.

To further enhance the flexibility of SeqPolicy in different working scenarios, a simple and computationally efficient pose estimation method is applied to estimate the object pose in the robot base frame directly without the need for calibration in each new scenario. One Siamense-CNN[45] is trained to generate representation mainly containing the object's pose information, which is used to estimate the hole's plane position. The estimated result is provided to the control policy directly as the observation.

2 Problem Statement

The multiple peg-in-hole assembly simulation environment is constructed in the Isaac Gym simulator as shown in Fig. 3. It contains the Franka Emika Panda robot with a Franka Emika two-finger gripper as the end effector, and objects (peg & hole) are

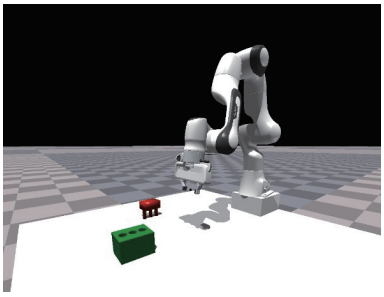

Fig. 3 Multiple peg-in-hole assembly scenario in simulation.

loaded on a white square table. The objects' three-dimensional model is depicted in Fig. 4.

In the simulation environment, the robot interacts with the objects over finite steps to pick the peg in the random initial state, align it with the hole, and insert it into the hole. Compared with the single peg-in-hole assembly task, this task has strict requirements for the object pose to successfully align and insert the peg into the hole. Moreover, the variation in the grasping point and the broader range of initial states for the objects create a more general and challenging environment compared to previous setups[5].

This task can be seen as apartially observable Markov decision process (POMDP). The control workflow Π is designed to sequentially execute the sub-tasks by maximizing the expected cumulative reward:

$$\Pi = \left\{\pi^*_{\text{pick}}, \pi^*_{\text{aligned}}, \pi^*_{\text{insert}}\right\} \tag{1}$$

$$\pi^* = \arg\max_{\pi} E_{\pi}\left[\sum_{t=0}^{N} \gamma^t r(s_t, a_t)\right] \tag{2}$$

where Π contains the control policies for executing the sub-tasks: picking the peg, aligning, and inserting the peg into the hole. N is a hyperparameter determining the maximum steps and the discount factor: $\gamma \in (0,1]$. During the training, the state in the time step t is denoted as s_t. Based on the action a_t taken at the time step, the reward $r(s_t, a_t)$ is obtained.

The objective is to effectively train the control policies for each sub-task, ensuring that the robot accurately completes this task.

3 Methodology

Several methods are utilized to implement this multiple peg-in-hole task. To simulate the real-world dynamic for policy learning and testing, domain randomization[46] is applied in the constructed simulation environment, including changing light and float physical parameters to ensure the robusticity of the control policy. The randomization parameters are listed in Table 1. The task is treated as a long-horizon task and is divided into three common sub-tasks: picking, alignment, and insertion. The PPO algorithm is applied to train the sequential control policies, guiding the robot to conduct the sub-tasks in sequence and implement the whole task. The task configuration is described in Section 3.1.

Observation is critical to the control policies, so an ablation experiment is conducted to test the different observations and the result is shown in Section 4.3. The most relevant modalities are objects' pose, proprioception, and force signal. Different sensors in each modality are combined to form the observation space in the experiment. Pose information is the most important, but obtaining an accurate pose is complex in real experiments. Therefore, the RGB image, which contains pose information and low-cost to collect, is an alternative. To extract the pose

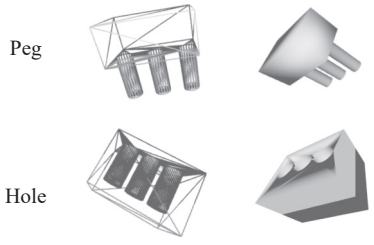

Fig. 4 Three-dimensional model of peg and hole.

information and improve sample efficiency, as described in Section 3.2, low-dimensional visual representation is learned from the high-dimensional RGB images using contrastive learning methods to estimate the objects' pose.

3.1 Reinforcement learning-based robotic control

Control policy and impedance controller are both applied to control the robots. The neural network in the control policy processes the collected observations and outputs the following states in Cartesian space for the robots. The impedance controller can avoid any collision or damage in case of some outlier actions from the network.

The action space in this study is 8-dimensional with the moving of position, orientation of the mid-point between two fingers, and the open/close signal of the gripper. Orientation is represented with quaternion. The action a_t in time step t is defined as the difference between the current and desired state:

$$a_t^{\text{pos}} = p_{t+1} - p_t, p = [x, y, z] \tag{3}$$

$$a_t^{\text{ori}} = q_{t+1} - q_t, q = q_0 + q_1 i + q_2 j + q_3 k \tag{4}$$

where t is the process's time step and $a_t^{\text{gri}} \in [0,1]$ is the state of the gripper, and 1 stands for open while 0 means close. p_t and q_t are the scalar values of the mid-point position and orientation in the time t. a_t^{pos} and a_t^{ori} are the outputs of the action by the control policy. Instead of directly outputting the predicted state, the distance between the predicted and current state is used in this study.

Observation space has 33 dimensions, including peg and hole pose: $X_{\text{peg}}, q_{\text{peg}}, X_{\text{hole}}, q_{\text{hole}}$, joint angles θ and forces F, and end-effector's pose $X_{\text{eef}}, q_{\text{eef}}$.

Proximal policy optimization (PPO): PPO is a popular algorithm, and it is applied in this study to train control policy with gradient clip and generalized advantage estimation (GAE) to

Algorithm 1 SeqPolicy for multiple peg-in-hole assembly

Initialize domain randomization parameters in simulator, policy network θ and value function network φ

for sub-tasks: picking, alignment, insert **do**

  Set the initial state

  repeat

    Observe state s_t, output action a_t for t timesteps

    Calculate joint actuation T

    Observe reward r_t and calculate reward A_t^{GAE}

    Update policy by maximizing PPO objective

    Fit value function on mean-squared error

  until Convergence to the controlled end-state

end for

make the policy learning robust. GAE is an exponentially weighted average of n-step returns with decay parameter λ:

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t) \tag{5}$$

$$A_t^{\mathrm{GAE}} = \sum_{l=0}^{T} (\gamma\lambda)^l \delta_{t+l} \tag{6}$$

where γ is the discount coefficient, r_t is the returned reward in time t and $V(s_t)$ is the state value in the state s_t. They are used to calculate the temporal difference (TD) error δ_t. When $\lambda = 0$, the return becomes the single-step return, which introduces low variance but high bias. While $\lambda = 1$ recovers the Monte-Carlo return G_t, bringing non-bias but high variance. The intermediate values of $\lambda \in (0,1)$ trade off the balance of the bias and variance in the training process.

To avoid the gradient explosion, the gradient clipping technique is used to limit the gradient to a reasonable bound. This prevents the new policy from being far from the old policy, keeping the training process stable. Therefore, the training objective can be written as:

$$\mathrm{Rto} = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t|s_t)} \tag{7}$$

$$\arg\max_\theta E_t\left[\min\left(\mathrm{Rto} * A_t^{\mathrm{GAE}}, \mathrm{clip}\left(\mathrm{Rto}, 1-\epsilon, 1+\epsilon\right) A_t^{\mathrm{GAE}}\right)\right] \tag{8}$$

where ϵ is a predefined parameter. The objective is to use a clipped probability ratio Rto to limit the change in the policy update and ensure the training is stable.

Considering the sub-tasks in this study are primitive skills that are simple and straightforward and do not require extensive exploration capabilities. Therefore, the PPO algorithm is suitable because of its stable performance during training and robustness across various environments. As shown in Algorithm 1, the SeqPolicy is trained with the PPO algorithm for all the sub-tasks: picking, alignment, and insertion.

3.2 Visual representation learning

The contrastive learning method is used in this paper to generate low-dimensional visual representations mainly containing pose information from high-dimensional RGB images. In this study, the objects are placed on the desk, so the object's height is settled with the desk's height, and pitch and roll are not a concern. Therefore, the objects' plane coordinates and yaw angles are recorded to identify dissimilarity between the two pose samples. By comparing these images, the neural network is trained to minimize distance among images with closer object poses and maximum distance among images with highly different object poses.

The Siamese network, including two similar backbone neural networks, is widely used for comparing inputs, which is suitable for contrastive learning[47,48]. In this study, a Siamese-CNN network is constructed with the same backbone neural networks. The RGB images are first converted to grayscale using the National Television System Committee (NTSC) formula. The backbone neural network then extracts the desired representation from the processed images. The training process and backbone network structure are shown in Fig. 5. The inputs are processed to output representations and compared to calculate similarity and loss for backpropagation. When reasoning, only one backbone network is activated. The structure of backbone networks is designed based on the VGG structure[49], which has been proven suitable for displacement recognition in images[45]. In this image, the Maxpool means the max pooling with stride two and 2×2 filter, and FC means the fully connected layer with the output dimension. The convolutional layer uses the same padding and one stride, its parameters are denoted as Conv<square kernel size>-<output channel number>, and the activation function is ReLU. The dataset is constructed with images collected in the simulation environment when the objects are in different positions and orientations. Figure 6 shows some examples.

This training process uses the information noise contrastive estimation (infoNCE) loss function to identify positive and negative samples. The scalar value 1 means the object's state in this image pair is identical, while 0 represents the biggest difference.

$$\mathcal{L}_{\mathrm{InfoNCE}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{\exp(\mathrm{sim}(z_i, z_j)/\tau)}{\sum_{k=1,k\neq i}^{N} \exp(\mathrm{sim}(z_i, z_k)/\tau)} \tag{9}$$

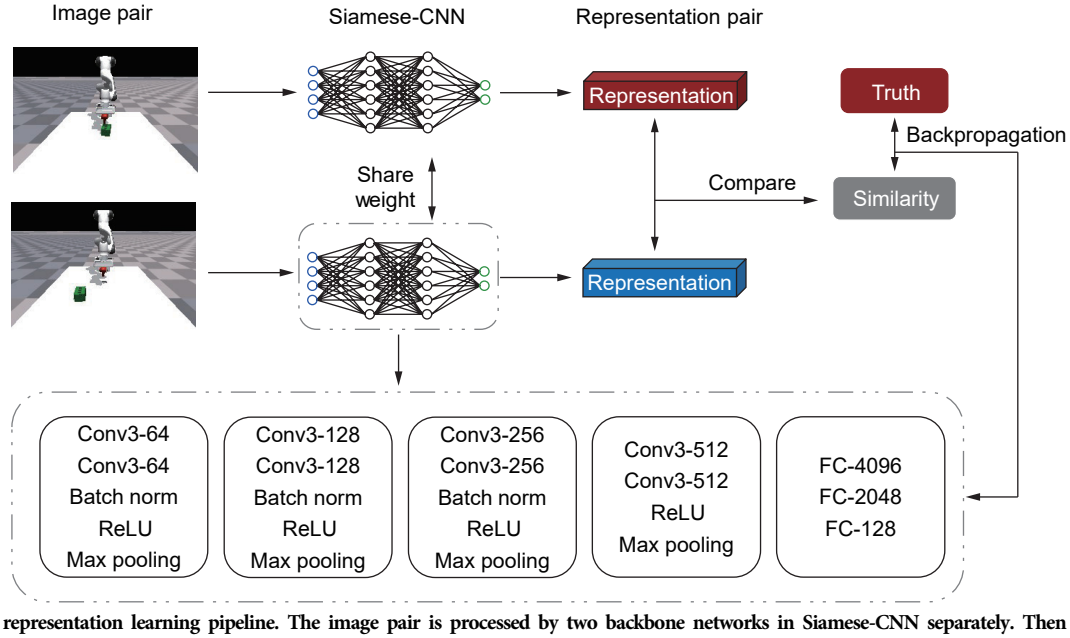

Fig. 5 Visual representation learning pipeline. The image pair is processed by two backbone networks in Siamese-CNN separately. Then a pair of 128-dimension visual representations is output, which can be directly used for the pose estimation task. By comparing these two representations, the estimated similarity is calculated. The loss is finally calculated based on ground truth and estimated similarity, which is used to update networks' parameters.

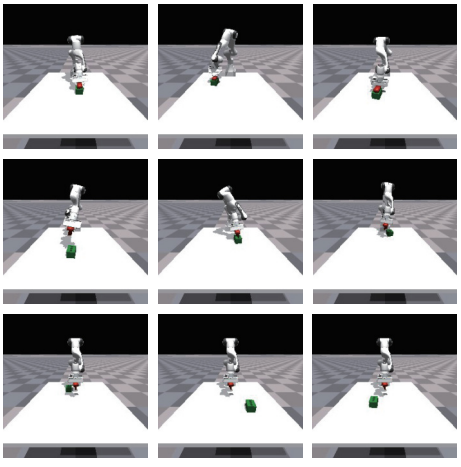

Fig. 6 Dataset examples. The dataset contains 2000 images captured from the side view, and their resolution is [128, 128, 3] (height, width, channel).

where N is the number of samples in the batch, including the positive pair and z_i is the representation vector of each sample. z_j is the positive sample representation vector, and z_k is the vector of other samples in the batch(i.e., the negative samples). τ is the temperature parameter, controlling the sharpness of the distribution. Lower values make the model focus more on harder negatives.

These representations are further used as input to a simple multilayer perceptron (MLP) network with two layers, which outputs the hole's plane coordinates, x, y and yaw angles. These coordinates are then provided to control policies as part of the observations.

4 Experiments

We validate the proposed method in this established simulation environment mentioned above. First, the training process and generalization test of sequential control policies are described. Then, the Siamese-CNN is trained with the image dataset, and the visual representation from the backbone network is used to output the hole's planar coordinates and rotation angle around the Z-axis. The pose information is sent to the control policy to implement the alignment and insertion task.

4.1 Experiment setup

To improve the adaptability of control policies, domain randomization is applied. The Gaussian noise is introduced to the sensor data, and several parameters are set dynamically, as listed in Table 1.

The objects are randomly spawned in the listed initial pose range with random physical parameters. When the gripper grasps, the grasp states where the fingers' pose on both sides of the peg

Table 1 Simulation environment variables.

Variable	Range
Grasp state	Uncertain
Object initial position	±0.18 m
Object initial rotation	±0.22 radian
Gravitational acceleration	±0.02 m/s^2
Mass	0.2–0.3
Joint stiffness	±0.1
Joint damping	±0.1

are uncertain. Other environmental parameters like gravitational acceleration and friction coefficient are also uncertain in each training epoch to simulate the dynamic world environment.

In the policy learning process, each episode contains 120 training steps, and it terminates when training steps over 120 or the success condition is satisfied. As mentioned before, this task has been divided into three sub-tasks, and the success conditions of each sub-task are picking up the peg, peg in a similar pose as the hole, and peg inside the hole.

The reward functions of each sub-task and single control policy are shown below.

Picking: Picking the peg and lifting it with a desired orientation and height.

$$D_{\mathrm{GP}} = \sqrt{(q_{\mathrm{eef}} - q_{\mathrm{peg}})^2 + (X_{\mathrm{eef}} - X_{\mathrm{peg}})^2} \tag{10}$$

$$r_{\mathrm{GP}} = 1 - \tanh(D_{\mathrm{GP}}) \tag{11}$$

Distance D_{GP} between the gripper fingers' midpoint and gripping points in the peg, and q_{eef} and q_{peg} are the quaternion of the gripper and peg separately, while X_{eef} and X_{peg} are the x, y, z Cartesian coordinate of gripper fingers' midpoint and peg. The reward r_{GP} increases while the distance D_{GP} decreases.

$$D_q = \sqrt{(q_{\mathrm{expect}} - q_{\mathrm{peg}})^2} \tag{12}$$

$$D_{\mathrm{height}} = |H_{\mathrm{peg}} - H_{\mathrm{expect}}| \tag{13}$$

$$r_G = 10 \times (1 - \tanh(0.6 \times D_{\mathrm{height}} + 0.4 \times D_q)) \tag{14}$$

$$r_{\mathrm{pick}} = \begin{cases} r_{\mathrm{GP}}, & \text{if } H_{\mathrm{peg}} = 0; \\ r_G + r_{\mathrm{GP}}, & \text{otherwise} \end{cases} \tag{15}$$

To control the end state of this sub-task, D_q and D_{height} are set to measure the distance between the current orientation q_{peg} and height H_{peg}, and the desired end orientation q_{expect} and height H_{expect}. The final reward r_{pick} is calculated based on the peg's situation.

Alignment: The peg should be aligned to the hole where the peg heading vertically to the hole and their center is closed.

$$D_{q'} = \sqrt{(q_{\mathrm{hole}} - q_{\mathrm{peg}})^2} \tag{16}$$

$$D_{\mathrm{center}} = \sqrt{(x_{\mathrm{hole}} - x_{\mathrm{peg}})^2 + (y_{\mathrm{hole}} - y_{\mathrm{peg}})^2} \tag{17}$$

$$r_{\mathrm{aligned}} = 10 \times (1 - \tanh(D_q + D_{\mathrm{center}})) \tag{18}$$

with smaller orientation distance of $D_{q'}$ and center point distance between peg and hole D_{center}, the reward r_{aligned} increases.

Insertion: When the peg is aligned with the hole, the robot should push the peg downward and insert the peg into the hole to finish the assembly task, the reward function for the insertion sub-task is defined below:

$$D_z = |z_{\mathrm{peg}} - z_{\mathrm{hole}}| \tag{19}$$

$$r_{\mathrm{insert}} = \begin{cases} 1 - \tanh D_z, & \text{if } D_{\mathrm{center}} < 0.01 \text{ and } D_{q'} < 0.02; \\ 0, & \text{otherwise} \end{cases} \tag{20}$$

where the reward r_{insert} increases when shorting the distance D_z of

the height of peg z_{peg} and hole z_{hole}, under the alignment condition.

Single control policy: There are two reward functions for the single control policy, parse, and dense reward function. The dense reward function simply combines all the reward functions in the previous sub-tasks. In the parse reward function, the reward can only be obtained when the success condition is achieved, that is, when a peg is inserted into the hole.

In the Isaac Gym, 8192 parallel environments are generated for training, and the entire training epochs are 1000. The collected data are states, observations, actions, and rewards from all the parallel environments. All the training hyperparameters are listed in Table 2.

4.2 Policy learning

As the definition of action space, observation space, and task in Section 3.1 and Section 4.1, the policy is trained in the simulation environment. The policy learning process of sequential control policies and single control policy is shown in Fig. 7.

According to the results, it is clear that the training process for sequential control policies is more effective than that of single control policy. In the training of control policy in each sub-task, the control policy converges quickly, achieving the goal quickly with around 150 epochs and keeping the performance stable. However, the training in single-policy learning is difficult and fails to achieve the ultimate goal.

There are two reward functions used in the single control policy training. The parse reward just sets the reward when the final goal, inserting the peg into the hole, is achieved, and the dense reward simply combines all the sub-task reward functions in sequential control policy. During the long training process, the exploration possibility is huge, and the sample efficiency is low, so it is difficult to find a solution with the sparse reward. The performance for dense reward is better, but the control policy is stuck in some sub-optimal situations, leading to failure in the entire training as the intermediate states are not clearly stated. For example, the peg keeps heading in the opposite direction of the hole, specifically heading top instead of heading down, or the peg drops from the gripper, and then the control policy keeps the gripper close and can never grasp the peg anymore, which happens around the 500 epochs in the single policy learning. In contrast, the configuration of the sub-tasks is simple and concise, and the PPO algorithm is stable and robust, so it has a satisfactory and stable performance.

Table 2 Policy leaning parameters.

Parameter	Value
Environments	8196
Batch size	16 384
Episode	120
Learning rate	5×10^{-4}
Reward scaling	1
Clip range	0.2
Discount factor	0.98
GAE factor	0.96

4.3 Abaltion experiment and generalization test

The observation space in the previous research varies. Still, they focus on three modalities: object pose, proprioception, and force feedback, and there are many sensors that could be used in each modality. To simplify the perception system and introduce less noise, this ablation experiment is conducted to explore a universal observation that provides enough information with fewer sensors in these multiple peg-in-hole assembly tasks. Considering the peg-in-hole task mainly focuses on precise insertion, in the following experiments, the control policy is trained to implement the align and insert task with different observations. After the trained control policy achieves the goal and keeps the performance stable, the success rate is calculated based on the successful insertion environments in a total of 8192 environments. The dimensions of different observations is shown in Table 3.

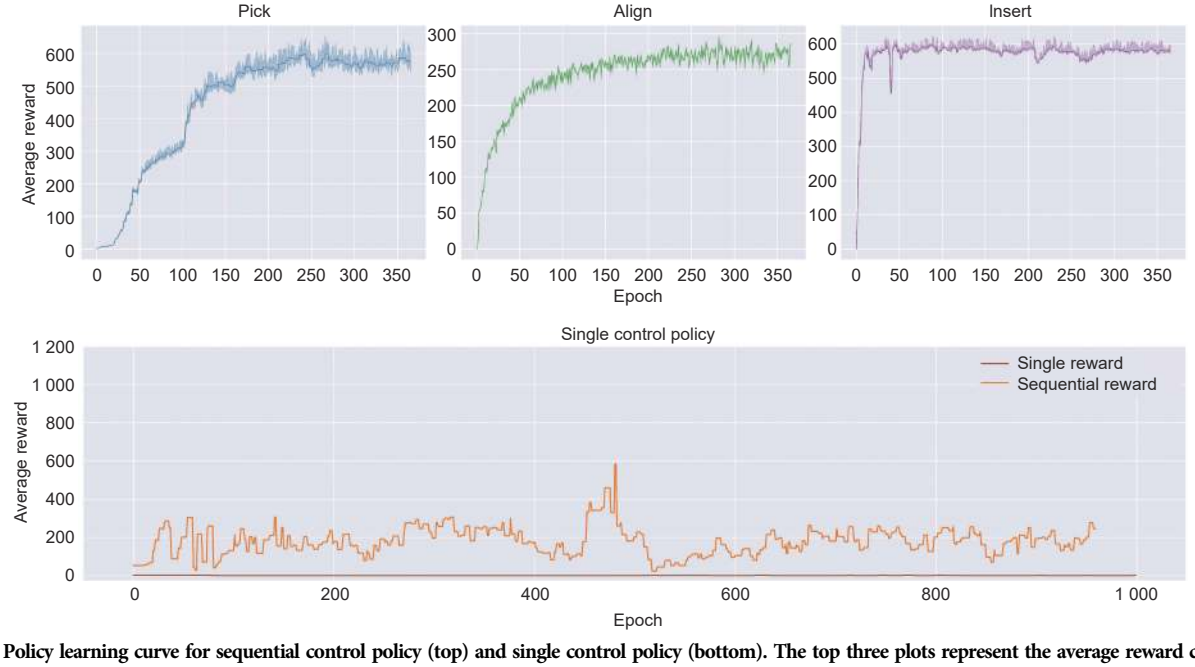

Fig. 7 Policy learning curve for sequential control policy (top) and single control policy (bottom). The top three plots represent the average reward curve in each training epoch of control policies for three sub-tasks. The bottom plot represents the best training process of a single control policy with two reward functions within the same epochs.

Table 3 Observation dimensions.

Observation	Dimension
{peg, end-effector, hole} pose	7
Joint angle	7
Hole position	3
Hand force	3

The pose includes the three-dimensional spatial position and orientation represented by a four-dimensional quaternion. The result is shown in Table 4. The better performance with more relevant information, but the improvement is not obvious after achieving an 80% success rate. Therefore, the joint angle, hole position, and hand joint force are combined to make a universal observation which has 13 dimensions. This observation can provide comprehensive information with less sensor data

To test the generalization ability of the control policies, one generalization experiment is conducted. The previously trained policy is applied directly to insert the pegs with different geometry into the hole. The sequential control policies' generalization ability is proven with success in all three intersecting surfaces: ellipse, square, and triangle. The insertion process is recorded in Fig. 8, and the success rate is shown in Table 5.

4.4 Pose estimation

As the training process described in Section 3.2, a Siamese-CNN is trained. Some results with processed images are shown in Fig. 9.

The results confirm that the visual representation effectively identifies the objects' pose. Therefore, the visual representation passes a three-layer MLP network to output the predicted hole's planar position. The prediction frequency could achieve 50Hz in the computer with GeForce RTX 2060 GPU and Intel i5-8300 CPU, which is sufficient for the real experiment.

Finally, in the policy learning process of the alignment sub-task, the visual representation is extracted from the images and used to estimate the hole's position online. This position is used as part of the observation in the training process, and the learning curve is shown in Fig. 10.

According to the learning curve in Figs. 7 and 10, the control policies with estimated and accurate pose information both achieve the goal. However, the learning process with estimated pose information is less effective. The instability in pose estimation brings vibration into policy learning, which causes a slower convergence speed and the held peg floating slightly, which should be stable. To exploit the full potential of the control policy, the explicit success criterion is not set in this training, and the reward scale is larger. Consequently, the reward climbs slowly over time. In fact, when the reward value exceeds 250, the control policy's performance is already satisfactory, although the gripper keeps waggling slightly.

Table 4 Ablation experiment result.

Observation	Sucess rate
Peg pose, end-effector pose, hole pose, hand force	0.912
Peg pose, hole position	0.898
Joint angle, hole position, hand force	0.882
End-effector pose, hole position	0.497
Joint angle, hole position	0.514
Joint angle, hand force	0.012

5 Conclusion and Discussion

This paper presents a solution for the multiple peg-in-hole assembly task, which can be transferred to other scenarios with varying object geometry, achieving an over 80% success rate. The uncertainties of grasping place, large-scale random initialization states, and domain randomization are incorporated into the simulation environment to improve the control policies' adaptability.

SeqPolicy demonstrates higher training efficiency and stable performance compared with the single control policy for long-term multiple peg-in-hole assembly task. During policy learning, SeqPolicy converges quickly around 200 epochs and maintains a high success rate across several runs. Additionally, the intermediate states, like lifting height, can be controlled based on expert experience, which is beneficial for improving sampling efficiency and flexible deployment in specific scenarios. To reduce the computational complexity, one ablation experiment investigates the compact universal observations: hole's planar position, joint angles, and hand joint force with 13 dimensions. Furthermore, the pose estimation method publishes the hole's pose at a high frequency, around 50 Hz, and assists the control policy in implementing the alignment task very well.

This pipeline implements the long-term peg-in-hole assembly task with a higher training efficiency, stable performance, and lower computational complexity. It can be flexibly applied to peg-in-hole assembly scenarios by fine-tuning the control policies with expert experience. However, in the real experiment, the observation should be adjusted based on the degrees of freedom of specific robots. For example, the Universal Robot has 6 degrees of freedom, while a Franka robot has 7 degrees of freedom.

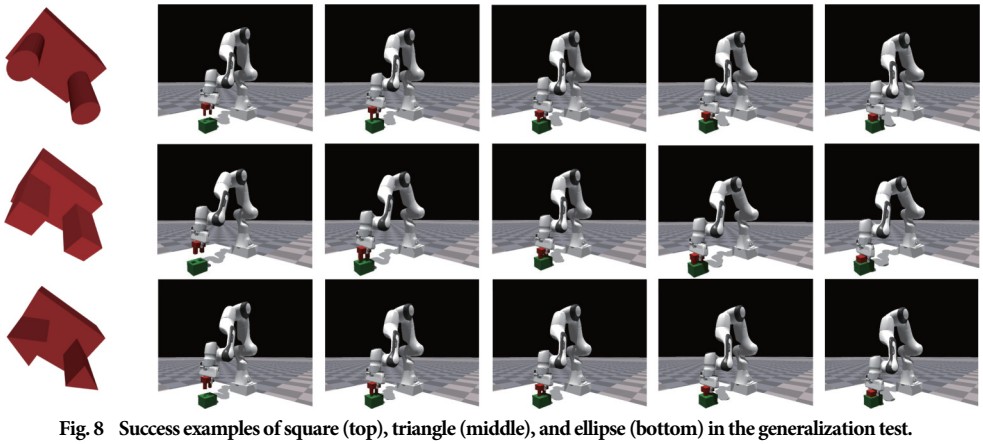

Fig. 8 Success examples of square (top), triangle (middle), and ellipse (bottom) in the generalization test.

Table 5 Success rate of generalization test.

Geometry	Success rate
Square	0.868
Triangle	0.787
Ellipse	0.901

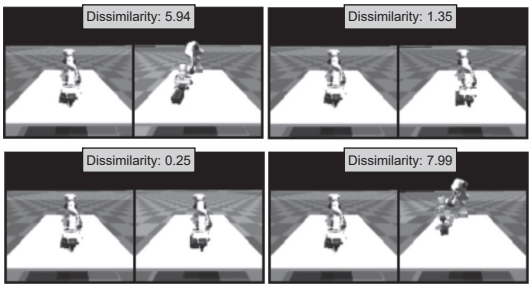

Fig. 9 Results of visual representation learning. The dissimilarity means how far the objects pose in the image pair.

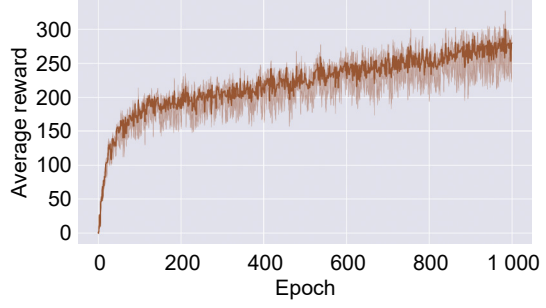

Fig. 10 Learning curve of alignment sub-task with estimated hole's pose.

This work explores a potential general framework for solving the multiple peg-in-hole assembly task with uncertainties of grasping location, object geometry, and initial working states. Its expansion potential is promising, especially with techniques in large models and visual representation. The large language model could be investigated to facilitate decision-making regarding the control policy based on the specific scenarios and human command (e.g., audio or textual) for automatic policy transition. Additionally, the visual representation could be improved using real-world datasets to estimate the object pose accurately in complex and unstructured environments, which could further improve the solution's adaptability in general scenarios, even including household scenarios.

Acknowledgment

This work was in part supported by the UKRI Guarantee funding for Horizon Europe MSCA Postdoctoral Fellowships (No. EP/Z00117X/1).

Article History

Received: 14 August 2024; Revised: 21 September 2024; Accepted: 20 October 2024

References

[1] K. P. Valavanis and K. M. Stellakis, A general organizer model for robotic assemblies and intelligent robotic systems, *IEEE Trans. Syst., Man, Cybern.*, vol. 21, no. 2, pp. 302–317, 1991.

[2] Y. Jiang, Z. Huang, B. Yang, and W. Yang, A review of robotic assembly strategies for the full operation procedure: Planning, execution and evaluation, *Robot. Comput. Integr. Manuf.*, vol. 78, p. 102366, 2022.

[3] C. C. Beltran-Hernandez, D. Petit, I. G. Ramirez-Alpizar, and K. Harada, Variable compliance control for robotic peg-in-hole assembly: A deep-reinforcement-learning approach, *Appl. Sci.*, vol. 10, no. 19, p. 6923, 2020.

[4] Y. Chen, C. Wang, F. F. Li, and K. Liu, Sequential dexterity: Chaining dexterous policies for long-horizon manipulation, in *Proc. 7th Conf. Robot Learning*, Atlanda, GA, USA, 2023, pp. 3809–3829.

[5] W. Chen, C. Zeng, H. Liang, F. Sun, and J. Zhang, Multimodality driven impedance-based Sim2Real transfer learning for robotic multiple peg-in-hole assembly, *IEEE Trans. Cybern.*, vol. 54, no. 5, pp. 2784–2797, 2024.

[6] H. Park, J. Park, D. H. Lee, J. H. Park, M. H. Baeg, and J. H. Bae, Compliance-based robotic peg-in-hole assembly strategy without force feedback, *IEEE Trans. Ind. Electron.*, vol. 64, no. 8, pp. 6299–6309, 2017.

[7] M. A. Lee, Y. Zhu, P. Zachares, M. Tan, K. Srinivasan, S. Savarese, F.-F. Li, A. Garg, and J. Bohg, Making sense of vision and touch: Learning multimodal representations for contact-rich tasks, *IEEE Trans. Robot.*, vol. 36, no. 3, pp. 582–596, 2020.

[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. Robotics and Biomimetics* (*ROBIO*), Kuala Lumpur, Malaysia, 2018, pp. 256–261.

[9] 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. Inf.*, vol. 15, no. 3, pp. 1658–1667, 2019.

[10] T. Inoue, G. De Magistris, A. Munawar, T. Yokoya, and R. Tachibana, Deep reinforcement learning for high precision assembly tasks, in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems* (*IROS*), Vancouver, Canada, 2017, pp. 819–825.

[11] H. Chen, G. Zhang, H. Zhang, and T. A. Fuhlbrigge, Integrated robotic system for high precision assembly in a semi-structured environment, *Assem. Autom.*, vol. 27, no. 3, pp. 247–252, 2007.

[12] J. Kober, J. A. Bagnell, and J. Peters, Reinforcement learning in robotics: A survey, *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1238–1274, 2013.

[13] A. Stooke, K. Lee, P. Abbeel, and M. Laskin, Decoupling representation learning from reinforcement learning, in *Proc. 38th Int. Conf. Machine Learning*, virtual, 2021, pp. 9870–9879.

[14] H. Nguyen and H. La, Review of deep reinforcement learning for robot manipulation, in *Proc. 3rd IEEE Int. Conf. Robotic Computing* (*IRC*), Naples, Italy, 2019, pp. 590–595.

[15] C. H. Wu and M. G. Kim, Modeling of part-mating strategies for automating assembly operations for robots, *IEEE Trans. Syst., Man. Cybern.*, vol. 24, no. 7, pp. 1065–1074, 1994.

[16] J. Xu, Z. Hou, Z. Liu, and H. Qiao, Compare contact model-based control and contact model-free learning: A survey of robotic peg-in-hole assembly strategies, arXiv preprint arXiv: 1904.05240, 2019.

[17] P. Falco, A. Attawia, M. Saveriano, and D. Lee, On policy learning robust to irreversible events: An application to robotic in-hand manipulation, *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 1482–1489, 2018.

[18] C. Zeng, S. Li, B. Fang, Z. Chen, and J. Zhang, Generalization of robot force-relevant skills through adapting compliant profiles, *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 1055–1062, 2022.

[19] X. Liu, Z. Liu, G. Wang, Z. Liu, and P. Huang, Efficient reinforcement learning method for multi-phase robot manipulation skill acquisition *via* human knowledge, *IEEE Trans. Automat. Sci. Eng.*, pp. 1–10, 2024.

[20] W. Tang, Y. Jiang, C. Zeng, H. Zhang, and H. Zhong, A reinforcement learning based control framework for robot gear assembly with demonstration learning and force feedback, in *Proc. 2024 IEEE Int. Conf. Industrial Technology* (*ICIT*), Bristol, UK, 2024, pp. 1–6.

[21] X. Liu, G. Wang, Z. Liu, Y. Liu, Z. Liu, and P. Huang, Hierarchical reinforcement learning integrating with human knowledge for practical robot skill learning in complex multi-stage manipulation, *IEEE Trans. Automat. Sci. Eng.*, vol. 21, no. 3, pp. 3852–3862, 2024.

[22] A. A. Apolinarska, M. Pacher, H. Li, N. Cote, R. Pastrana, F. Gramazio, and M. Kohler, Robotic assembly of timber joints using reinforcement learning, *Autom. Constr.*, vol. 125, p. 103569, 2021.

[23] T. Chen, J. Xu, and P. Agrawal, A system for general in-hand object re-orientation, in *Proc. 5th Conf. Robot Learning*, London, UK, 2021, pp. 297–307.

[24] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto, Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation, in *Proc. IEEE Int. Conf. Robotics and Automation* (*ICRA*), London, UK, 2023, pp. 5954–5961.

[25] R. S. Sutton, D. Precup, and S. Singh, Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning, *Artif. Intell.*, vol. 112, no. 1–2, pp. 181–211, 1999.

[26] L. P. Kaelbling and T. Lozano-Perez, Hierarchical task and motion planning in the now, in *Proc. IEEE Int. Conf. Robotics and Automation*, Shanghai, China, 2011, pp. 1470–1477.

[27] C. Agia, T. Migimatsu, J. Wu, and J. Bohg, STAP: Sequencing task-agnostic policies, in *Proc. IEEE Int. Conf. Robotics and Automation* (*ICRA*), London, UK, 2023, pp. 7951–7958.

[28] T. Kipf, Y. Li, H. Dai, V. Zambaldi, A. Sanchez-Gonzalez, E. Grefenstette, P. Kohli, and P. Battaglia, CompILE: compositional imitation learning and execution, in *Proc. 36th Int. Conf. Machine Learning*, Long Beach, CA, USA, 2019, pp. 3418–3428.

[29] G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto, Robot learning from demonstration by constructing skill trees, *Int. J. Robot. Res.*, vol. 31, no. 3, pp. 360–375, 2012.

[30] C. Wang, L. Fan, J. Sun, R. Zhang, F. F. Li, D. Xu, Y. Zhu, and A. Anandkumar, MimicPlay: long-horizon imitation learning by watching human play, arXiv preprint arXiv: 2302.12422, 2023.

[31] O. Nachum, S. S. Gu, H. Lee, and S. Levine, Data-efficient hierarchical reinforcement learning, in *Proc. 32nd Conf. Neural Information Processing Systems* (*NeurIPS 2018*), Montréal, Canada, 2018, pp. 3307–3317.

[32] J. Oh, S. Singh, H. Lee, and P. Kohli, Zero-shot task generalization with multi-task deep reinforcement learning, in *Proc. 34th Int. Conf. Machine Learning*, Sydney, Australia, 2017, pp. 2661–2670.

[33] T. D. Kulkarni, K. R. Narasimhan, A. Saeedi, and J. B. Tenenbaum, Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation, in *Proc. 30th Int. Conf. Neural Information Processing Systems* (*NIPS 2016*), Barcelona, Spain, 2016, pp. 3682–3690.

[34] D. Han, B. Mulyana, V. Stankovic, and S. Cheng, A survey on deep reinforcement learning algorithms for robotic manipulation, *Sensors*, vol. 23, no. 7, p. 3762, 2023.

[35] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, Proximal policy optimization algorithms, arXiv preprint arXiv: 1707.06347, 2017.

[36] G. Konidaris and A. Barto, Skill discovery in continuous reinforcement learning domains using skill chaining, in *Proc. 23rd Int. Conf. Neural Information Processing Systems* (*NIPS 2013*), Vancouver, Canada, 2013, pp. 1015–1023.

[37] X. B. Peng, M. Chang, G. Zhang, P. Abbeel, and S. Levine, MCP: Learning composable hierarchical control with multiplicative compositional policies, in *Proc. 33rd Conf. Neural Information Processing Systems* (*NeurIPS 2019*), Vancouver, Canada, 2019, pp. 3686–3697.

[38] Y. Lee, J. J. Lim, A. Anandkumar, and Y. Zhu, Adversarial skill chaining for long-horizon robot manipulation *via* terminal state regularization, arXiv preprint arXiv: 2111.07999, 2021.

[39] Y. Bengio, A. Courville, and P. Vincent, Representation learning: A review and new perspectives, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.

[40] A. Spurr, A. Dahiya, X. Wang, X. Zhang, and O. Hilliges, Self-supervised 3D hand pose estimation from monocular RGB *via* contrastive learning, in *Proc. 2021 IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Nashville, TN, USA, 2021, pp. 11230–11239.

[41] Y. Xiao, Y. Du, and R. Marlet, PoseContrast: Class-agnostic object viewpoint estimation in the wild with pose-aware contrastive learning, in *Proc. Int. Conf. 3D Vision* (*3DV*), London, UK, 2021, pp. 74–84.

[42] A. Khandelwal, L. Weihs, R. Mottaghi, and A. Kembhavi, Simple but effective: CLIP embeddings for embodied AI, in *Proc. 2022 IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), New Orleans, LA, USA, 2022, pp. 14829–14838.

[43] F. Liu, F. Yan, L. Zheng, C. Feng, Y. Huang, and L. Ma, RoboUniView: visual-language model with unified view representation for robotic manipulation, arXiv preprint arXiv: 2406.18977, 2024.

[44] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark et al., Learning transferable visual models from natural language supervision, in *Proc. 38th Int. Conf. Machine Learning*, virtual, 2021, pp. 8748–8763.

[45] X. Liu, Z. Rozsypálek, and T. Krajník, Self-supervised learning for fusion of IR and RGB images in visual teach and repeat navigation, in *Proc. European Conf. Mobile Robots* (*ECMR*), Coimbra, Portugal, 2023, pp. 1–7.

[46] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, et al., Isaac Gym: High performance GPU-based physics simulation for robot learning, in *Proc. 35th Conf. Neural Information Processing Systems Track on Datasets and Benchmarks*, virtual, 2021.

[47] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, Momentum contrast for unsupervised visual representation learning, in *Proc. 2020 IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Seattle, WA, USA, 2020, pp. 9726–9735.

[48] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, A simple framework for contrastive learning of visual representations, in *Proc. 37th Int. Conf. Machine Learning*, virtual, 2020, pp. 1597–1607.

[49] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv: 1409.1556, 2014.