Parallax-based Imitation Learning with Human Intervention for Uncertain Insertion Tasks

Yasuharu Niwa[1], Kunihito Kato[1],
Hiroaki Aizawa[2], Yoshiyuki Hatta[1] and Kazuaki Ito[1]

[1] Gifu University, 1-1 Yanagito, Gifu City, Gifu 501-1193, Japan
[2] Hiroshima University, 1-3-2 Kagamiyama, Higashi-Hiroshima City,
Hiroshima 739-8511, Japan
kkato@gifu-u.ac.jp

Abstract. Standard insertion machines require pre-determined component position and posture. If the position and posture change every time, we must solve this problem. Most conventional methods attempted to solve this task by identifying the position and posture. However, these methods require a multi-step strategy following the handmade rule. This paper proposes an imitation learning method to automate the wire insertion task with uncertainties in position and posture. The proposed model learns the motion policy through human demonstrations and maps image data to the robot's action in a single step. Moreover, the model considers the parallax of the stereo images for accurate insertion. In addition, the model outputs the insertion action and recovery action to recover from insertion failures. However, the standard data collection method cannot collect recovery actions, and manual labeling of action classes is essential. This paper proposes a novel data collection method called "Labeling with Human Intervention (LHI)" to tackle this problem. This method automatically generates action labels and collects recovery action with human intervention. We conducted real-space insertion tests and found that our approach achieved 96.3% (104/108).

Keywords: Deep Imitation Learning, Image Processing, Robotics.

1 Introduction

Standard insertion machines based on force control [1] require pre-determined component position and posture. If the position and posture change every time, we must solve this problem. Most conventional methods focused on the uncertainty of the hole position or the component posture. However, these methods require multiple steps through pre-determined rules to identify the position or posture.

In this paper, we apply imitation learning [2] to automate the insertion task shown in Fig 1, where both the hole position and the wire posture are random. In imitation learning, the model learns the motion policy through human demonstrations. The

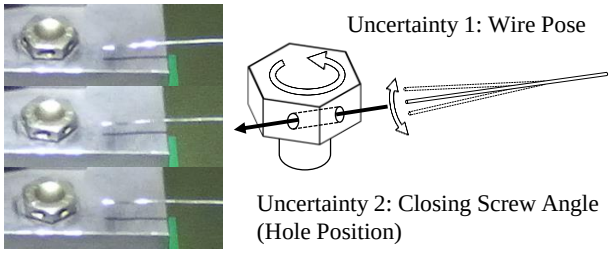


Fig. 1. Insertion Task with Two Uncertainties.

trained model can directly map the input data to the robot's action in a single step without object detection and pose estimation. This paper automates the insertion task via imitation learning using a single model and a single stereo camera.

Our proposed model considers the parallax of stereo images in the feature extractor for accurate wire insertion. In addition, the model outputs insertion action and recovery action to recover from insertion failures. However, the standard data collection method cannot collect the recovery action. Moreover, it is necessary to label action classes manually. This paper proposes a novel data collection method called "Labeling with Human Intervention (LHI)" to tackle this problem. The weak model inserts the wire, and the operator intervenes if the insertion fails. The proposed method collects recovery actions through human intervention and automatically gives action labels. Furthermore, this paper proposes a novel noise added to human actions to expand the input data distribution and reduce the instability of motion generation.

We conducted real-space insertion tests and found that our approach achieved 97.2% (35/36) and improved by 11.1% from the baseline model. Finally, we evaluated our method in a more complex environment with no blackout curtains and automatic grasping, and the success rate was 96.3% (104/108). The results showed that our method could recover from insertion failures.

2 Related Work

2.1 Autonomous Insertion

Previous works focus on the hole position uncertainty or the component posture one. The methods that tackle the hole position insert the component using images [3] or force [4, 5]. These approaches use an object detection model, like YOLOv3 [6], or force sensors attached to the robot to detect the hole position. The methods that tackle the component posture use images [7, 8] or tactile [9]. Image-based methods use contour extraction, triangulation [7], or image processing with deep learning [8]. Tactile-based methods use tactile sensors attached to the robot's fingers to detect the component posture [9]. These methods [3-9] require multi-step processes following pre-determined rules. Another issue is applying completely different approaches for the position and posture. In contrast, this paper tackles these uncertainties using only one model and one stereo camera.

2.2 Imitation Learning

Imitation learning [2] is a method for making autonomous robots without complex multi-step strategies. This method learns the motion policy using human demonstrations. The trained model directly maps the input data to the robot's action. Our approach exploits and extends the basic principle of imitation learning to omit hole detection or pose estimation.

Previous works in imitation learning have achieved complex manipulations, such as grasping a fish [13], needle threading [15], and peeling bananas [18]. Most studies use images [10-18] or force/position data [19-25] as the primary input to generate robot motion. In this paper, we select the stereo images as the input data. In image-based methods, the operator monitors the robot's state using an HMD (Head Mounted Display) and moves the robot remotely. The HMD presents binocular images to the operator, who can recognize the robot's space remotely and three-dimensionally [14-18]. This paper uses two RGB cameras and presents the stereo images to the operator.

Recent works use the linear velocity of the robot's tip as the model output [11-12, 14-18]. The model generates a trajectory by adding up the output of each time step. However, if the output is a single linear velocity, the robot cannot recover from insertion failure [15]. Therefore, our model outputs an insertion action and a recovery action following the model proposed by H. Kim et al. [15]. However, the method [15] needs to label action classes manually for the autonomous action selection. This paper proposes a novel data collection method LHI to avoid manual labeling.

2.3 Covariate Shift

There is a problem called "covariate shift" in the machine learning field. The covariate shift is when the input distribution changes in training and testing. The model inputted unseen distribution data cannot predict appropriate values. In imitation learning, the robot enters unseen situations if the policy model generates wrong actions. Moreover, the input distribution is different from the training data, and the policy model repeats to output incorrect actions. Previous works [26-29] try to expand the input distribution in human demonstrations to solve the covariance shift. In an early study, DAgger (Dataset Aggregation) [26] extends the input distribution by randomly switching between the policy model and the operator. However, this approach has a problem: the operator must keep monitoring the robot until getting a well-learned model. The recent method, DART (Disturbances for Augmenting Robot Trajectories) [27], injects noise into the operator's actions to enlarge the input distribution efficiently. This paper uses the noise injection method to expand training data distribution. The studies about the types of noise change the robot's initial position in each episode [28] or add randomly triangular noise to the operator's steering action [29]. In our method, we change the robot's initial position and add the novel triangular noise improved to three dimensions using a rotation matrix.

3 Method

3.1 Fundamental Principle

The imitation learning model learns the motion policy through the demonstration set $\mathcal{D} = \{\mathcal{C}^i\}_{i=1}^{N}$. N is the number of trials. $\mathcal{C}^i = \{(\boldsymbol{s}^t, \boldsymbol{a}^t)\}_{t=1}^{L}$ is the demonstration represented as the set of combinations of the robot's state $\boldsymbol{s}^t$ and the operator's command $\boldsymbol{a}^t$ at each time t. L is the end time of each demonstration. The model has learnable parameters optimized using standard supervised learning with the input $\boldsymbol{s}^t$ and target signal $\boldsymbol{a}^t$. The well-learned model outputs the robot's action like the operator's command. Therefore, the model can move the robot instead of the operator. This paper makes the robot insert the wire, expanding this fundamental principle.

3.2 Model Architecture

The proposed model expands the baseline model [15] that outputs insertion and recovery action. Fig. 2 (a) shows the architecture of the baseline model. Fig. 2 (b) shows the details of each module. The baseline model has a standard CNN and requires the concatenation of stereo images. Then, the model transmits the feature to the insertion module, recovery module, recovery classifier, and recovery step predictor. The insertion module outputs a linear velocity $\boldsymbol{o}_{\mathrm{ins}}^t$ to insert the wire into the hole. The recovery module outputs another linear velocity $\boldsymbol{o}_{\mathrm{rec}}^t$ to pull back the wire. The recovery classifier outputs a probability o_{label}^t to select the appropriate action. The recovery step predictor outputs a consecutive step length o_{step}^t of the recovery action to avoid stagnating the wire. This paper calls the last three modules sub-modules.

Fig. 3 shows the proposed model. There are two improvements over the baseline model. First, the proposed model uses CNN^2 [30]. The CNN^2 feature extractor incorporates both binocular and monocular stereoscopic information. Fig. 4 shows the CNN^2 architecture. The DFP (Dual Feedforward Pathways) block [30] concatenates the differences between left and right features to consider the parallax-based distance perspective. The CM (Concentric Multi-Scale) pooling [30] conducts the multi-scale pooling and connects the outputs in the channel axes to consider the blurry vision caused by distance.

This paper introduces an LSTM layer in the recovery classifier in the second improvement. As shown in Fig. 5, there are two failure patterns in the wire insertion task. The first failure is passing through the outside of hole. The second failure is the bending of the wire with a collision with the screw face. It is crucial to consider the positional relation between the wire and the hole to recover from the first failure. Thus, it is sufficient for the recovery classifier to receive the one-frame image feature. In contrast, it is necessary to consider time series features for wire deformation failure. Therefore, our recovery classifier has an LSTM layer to memorize past wire information.

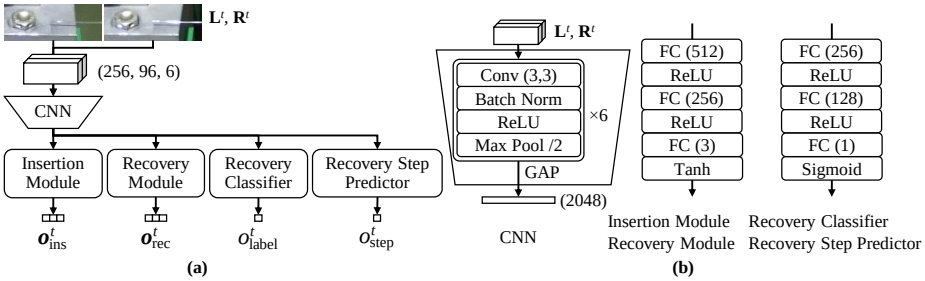


Fig. 2. Baseline Model

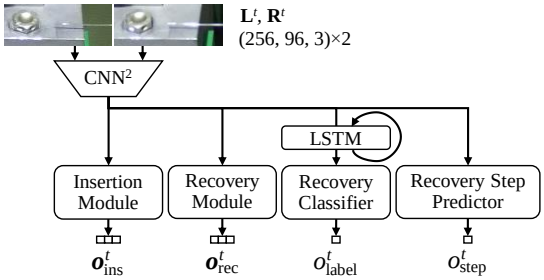


Fig. 3. Proposed Model

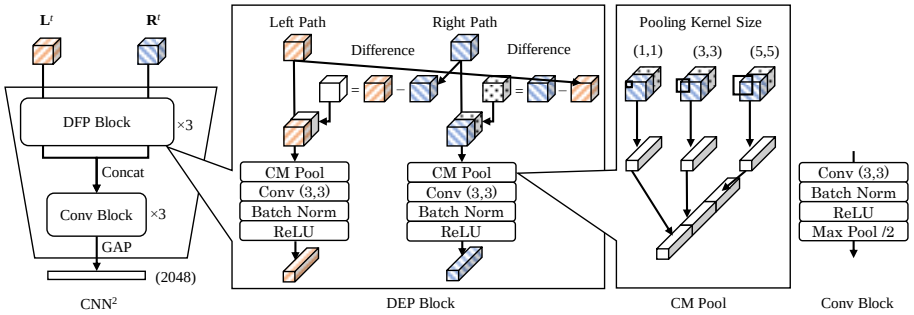


Fig. 4. Feature Extractor CNN^2

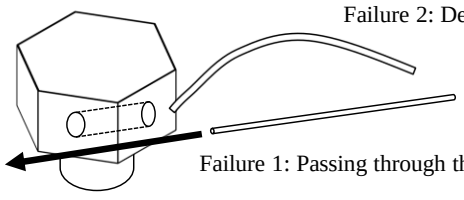


Fig. 5. Failure Patterns of Wire Insertion.

3.3 Collecting Insertion Action

This section explains the first step in our approach to collect the operator's insertion action. Fig. 6 shows the loop for collecting the insertion action, and Algorithm 1 shows the details. At first, our system randomly determines the starting position of the slave robot to augment the input distribution. After the slave robot moves to the start position, the operator inserts the wire into the hole remotely. The stereo camera gets binocular images cropped to remove unnecessary information at each time t. The operator watches the slave robot's state remotely and three-dimensionally through the cropped images. The HMD view is the same as the model inputs. Therefore, the operator can confirm that the images have the information for motion generation.

In addition, the operator inputs the linear velocity $\boldsymbol{v}^t$ using the master robot, and the system multiplies a rotation matrix $\mathbf{M}^t$ to the action $\boldsymbol{v}^t$ to expand the input distribution. $\mathbf{M}^t$ has a function to rotate the direction of $\boldsymbol{v}^t$, and the system randomly determines the angles using triangular noise [29]. Fig. 7 shows the steps to inject the noise. Fig. 8 shows the two triangular noises α^t and β^t, which have a random amplitude, length, and occurrence probability. The system calculates the rotated action $\tilde{\boldsymbol{v}}^t = \mathbf{M}^t\boldsymbol{v}^t$ and updates the slave robot's position by adding up each time output $\tilde{\boldsymbol{v}}^t$. At each teleoperation loop, our system saves the combinations of the stereo image and the action $((\mathbf{L}^t, \mathbf{R}^t), \boldsymbol{v}^t)$. At last, we get the demonstration set $\mathcal{D}_{\text{ins}}$, including the operator's insertion actions.

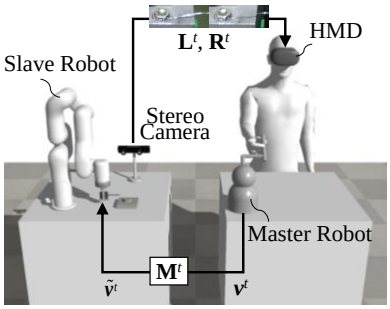


Fig. 6. Manual Mode

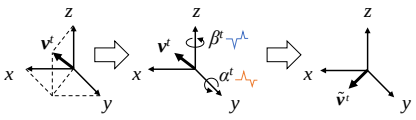


Fig. 7. Rotation of Human Actions

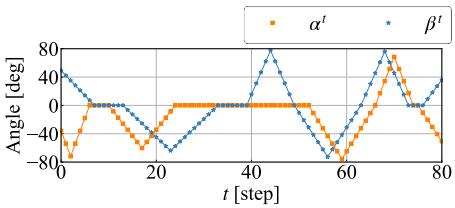


Fig. 8. Triangular Noise

Algorithm 1: Manual Mode

Parameter: Home position $\boldsymbol{p}_{\text{home}}$, positional noise $\boldsymbol{p}_{\text{noise}}$, slave robot position $\boldsymbol{p}^t$, x-coordinate of the slave robot p_x^t, x-coordinate of the end position p_{end}, trial number N, time $t \leftarrow 0$, demonstration set $\mathcal{D}_{\text{ins}} \leftarrow \{\}$, demonstration $\mathcal{C}_{\text{ins}}^i \leftarrow \{\}$.

1. **for** $i = 1$ **to** N **do**
2. $\quad \boldsymbol{p}_{\text{noise}} \leftarrow$ random values.
3. $\quad \boldsymbol{p}^t \leftarrow \boldsymbol{p}_{\text{home}} + \boldsymbol{p}_{\text{noise}}$.
4. $\quad t \leftarrow 0$.
5. $\quad \mathcal{C}_{\text{ins}}^i \leftarrow \{\}$.
6. $\quad$ **while** $p_x^t < p_{\text{end}}$ **do**
7. $\quad\quad$ Get $(\mathbf{L}^t, \mathbf{R}^t)$ from the camera.
8. $\quad\quad$ Send $(\mathbf{L}^t, \mathbf{R}^t)$ to the HMD.
9. $\quad\quad$ Get $\boldsymbol{v}^t$ of the master robot.
10. $\quad\quad$ Get the rotation matrix $\mathbf{M}^t$.
11. $\quad\quad \tilde{\boldsymbol{v}}^t = \mathbf{M}^t\boldsymbol{v}^t$.
12. $\quad\quad \boldsymbol{p}^t \leftarrow \boldsymbol{p}^t + \tilde{\boldsymbol{v}}^t$.
13. $\quad\quad \mathcal{C}_{\text{ins}}^i \leftarrow \mathcal{C}_{\text{ins}}^i \cup \{((\mathbf{L}^t, \mathbf{R}^t), \boldsymbol{v}^t)\}$.
14. $\quad\quad t \leftarrow t + 1$.
15. $\quad$ **end while**
16. $\quad \mathcal{D}_{\text{ins}} \leftarrow \mathcal{D}_{\text{ins}} \cup \mathcal{C}_{\text{ins}}^i$.
17. **end for**
18. **Return** $\mathcal{D}_{\text{ins}}$.

3.4 Collecting Recovery Action

In the second step, we collect the recovery action through LHI. Algorithm 2 indicates the details. Fig. 9 shows the LHI loop, in which the operator and a weak model control the slave robot while switching from one to another. The right loop in Fig. 9 indicates the operator control, and the left shows the weak model control. This method first prepares the weak model with the feature extractor and the insertion module trained using $\mathcal{D}_{\text{ins}}$ for one epoch. Then, the weak model operates the slave robot. If the weak model falls into errors like Fig. 5, the operator intervenes using the master robot button and pulls back the wire. At this step, the system saves the operator's action as the recovery action and the weak model action as the insertion action. The previous method [15] made the action labels by hand. In contrast, our method can automatically provide them. At last, our system labels the recovery steps for training the recovery step predictor, and we get the demonstration set $\mathcal{D}_{\text{rec}}$, including the operator's recovery actions.

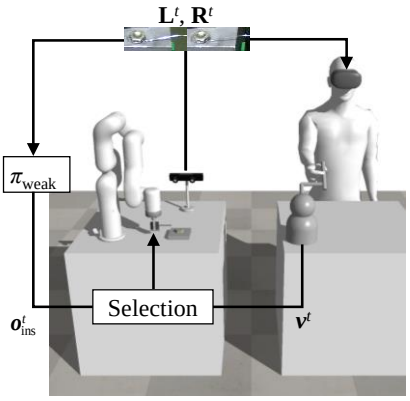


Fig. 9. Semi-autonomous Mode

Algorithm 2: Semi-autonomous Mode (LHI)

Parameter: Home position $\boldsymbol{p}_{\text{home}}$, positional noise $\boldsymbol{p}_{\text{noise}}$, slave robot position $\boldsymbol{p}^t$, x-coordinate of the slave robot p_x^t, x-coordinate of the end position p_{end}, trial number N, time $t \leftarrow 0$, demonstration set $\mathcal{D}_{\text{rec}} \leftarrow \{\}$, demonstration $\mathcal{C}^i_{\text{rec}} \leftarrow \{\}$, weak model $\pi_{\text{weak}}(\cdot)$, the button of the master robot $b^t \leftarrow$false, action label $c^t \leftarrow 0$.

for $i = 1$ **to** N **do**
  $\boldsymbol{p}_{\text{noise}} \leftarrow$ random values.
  $\boldsymbol{p}^t \leftarrow \boldsymbol{p}_{\text{home}} + \boldsymbol{p}_{\text{noise}}$.
  $t \leftarrow 0$.
  $\mathcal{C}^i_{\text{rec}} \leftarrow \{\}$.
  while $p_x^t < p_{\text{end}}$ **do**
    Get $(\mathbf{L}^t, \mathbf{R}^t)$ from the camera.
    Send $(\mathbf{L}^t, \mathbf{R}^t)$ to the HMD.
    Get $\boldsymbol{v}^t$ of the master robot.
    $\boldsymbol{o}^t_{\text{ins}} \leftarrow \pi_{\text{weak}}(\mathbf{L}^t, \mathbf{R}^t)$.
    if b^t **then**
      $\boldsymbol{p}^t \leftarrow \boldsymbol{p}^t + \boldsymbol{v}^t$.
      $c^t \leftarrow 1$.
    else
      $\boldsymbol{p}^t \leftarrow \boldsymbol{p}^t + \boldsymbol{o}^t_{\text{ins}}$.
      $c^t \leftarrow 0$.
    end if
    $\mathcal{C}^i_{\text{rec}} \leftarrow \mathcal{C}^i_{\text{rec}} \cup \{((\mathbf{L}^t, \mathbf{R}^t), \boldsymbol{v}^t, c^t)\}$.
    $t \leftarrow t + 1$.
  end while
  $\mathcal{D}_{\text{rec}} \leftarrow \mathcal{D}_{\text{rec}} \cup \mathcal{C}^i_{\text{rec}}$.
end for
Get recovery steps $\mathcal{L}$.
$\mathcal{D}_{\text{rec}} \leftarrow \mathcal{D}_{\text{rec}} \cup \mathcal{L}$.
Return $\mathcal{D}_{\text{rec}}$.

3.5 Autonomous Insertion

In the third step, the slave robot insets the wire autonomously. Fig. 10 shows the autonomous control loop, and Algorithm 3 indicates the details. The process is the same as the previous algorithm [15], excepting the initialization of the LSTM. This method first prepares the model, including the sub-modules, trained using $\mathcal{D}_{\text{ins}}$ and $\mathcal{D}_{\text{rec}}$. The well-learned model inserts the wire into the hole with occasional recovery.

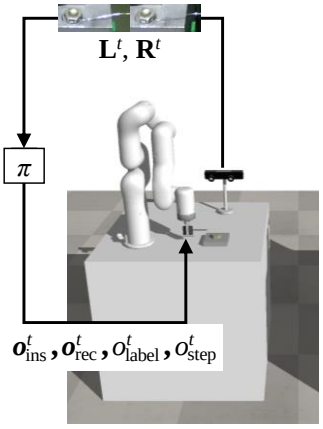


Fig. 10. Autonomous Mode

Algorithm 3: Autonomous Mode

Parameter: Home position $\boldsymbol{p}_{\text{home}}$, slave robot position $\boldsymbol{p}^t$, x-coordinate of the slave robot p^t_x, x-coordinate of the end position p_{end}, trial number N, time $t \leftarrow 0$, model $\pi(\cdot)$, selected action $\boldsymbol{o}^t$, threshold c_{th}, recovery steps $k \leftarrow 0$.

1. **for** $i = 1$ **to** N **do**
2. $\quad \boldsymbol{p}^t \leftarrow \boldsymbol{p}_{\text{home}}$.
3. $\quad t \leftarrow 0$
4. $\quad$ **while** $p^t_x < p_{\text{end}}$ **do**
5. $\quad\quad$ Get $(\mathbf{L}^t, \mathbf{R}^t)$ from the camera.
6. $\quad\quad \boldsymbol{o}^t_{\text{ins}}, \boldsymbol{o}^t_{\text{rec}}, o^t_{\text{label}}, o^t_{\text{step}} \leftarrow \pi(\mathbf{L}^t, \mathbf{R}^t)$.
7. $\quad\quad \boldsymbol{o}^t \leftarrow \boldsymbol{o}^t_{\text{ins}}$
8. $\quad\quad$ **if** $o^t_{\text{label}} > c_{\text{th}}$ **and** $k = 0$ **then**
9. $\quad\quad\quad k \leftarrow o^t_{\text{step}}$
10. $\quad\quad$ **end if**
11. $\quad\quad$ **if** $k > 0$ **then**
12. $\quad\quad\quad$ Initialize LSTM.
13. $\quad\quad\quad \boldsymbol{o}^t \leftarrow \boldsymbol{o}^t_{\text{rec}}$
14. $\quad\quad\quad k \leftarrow k - 1$.
15. $\quad\quad$ **end if**
16. $\quad\quad \boldsymbol{p}^t \leftarrow \boldsymbol{p}^t + \boldsymbol{o}^t$.
17. $\quad\quad t \leftarrow t + 1$.
18. $\quad$ **end while**
19. **end for**

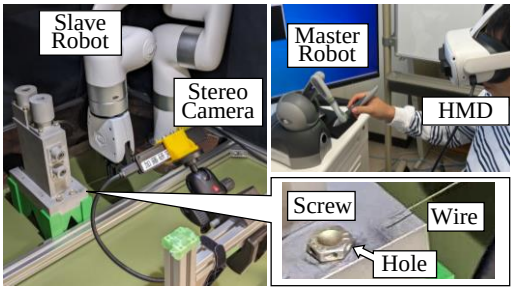


Fig. 11. Equipment

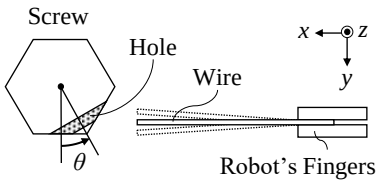


Fig 12. Screw Angle and Wire Pose

4 Experiments

4.1 Equipment

Fig. 11 shows the experimental equipment using the slave robot xArm6, the stereo camera ZED mini, the master robot Geomagic Touch, and HMD Oculus Quest 2. The diameter of the hole is 1.8 mm. The diameter of the wire is 0.8mm. Fig. 12 shows the random range of the screw angle θ and the wire tip. The angle range is [0,30] degrees. The wire tip changes within [−5,5] mm in the axes of y and z.

4.2 Dataset

We collected $\mathcal{D}_{\text{ins}}$ by inserting the wire 600 times and randomly divided the data into three parts, 39,248 training data, 3,584 validation data, and 616 test data. Then, we split $\mathcal{D}_{\text{rec}}$ from 300 trials in Algorithm 2 into 15,899 training data, 1,570 validation data, and 344 test data.

4.3 Assessment of Feature Extractor

This paper first compares the success rates without the recovery action. We train the feature extractor and the insertion module using $\mathcal{D}_{\text{ins}}$. The baseline model uses a standard 6-layer CNN. In contrast, the proposed model uses a CNN^2. We optimize these models using Adam (learning rate 10^{-6}, batch size 32) and RMSE and select the weights with the smallest validation loss in 50 epochs. We evaluate 36 insertions with a variety of hole positions and wire postures. Table 1 shows the success rates. The failure that the wire passed through outside of the hole was the most frequent, with nine failures in the baseline model and four in the proposed model. The proposed model using CNN^2, considering the sense of distance, was able to align the wire tip more accurately than the baseline model.

Table 1. Success Rate without Sub-modules

Policy Model (No Sub-modules)	Success Rate %	Number of Successes times
Baseline Model [15]	75.0	27/36
Proposed Model	83.3	30/36

4.4 Assessment of Recovery Classifier

This section compares the accuracies of the recovery classifier. The feature extractors and the insertion modules have the weights obtained in section 4.3. In contrast, we train the sub-modules using $\mathcal{D}_{\text{rec}}$. We use Adam (learning rate 10^{-5}, batch size 32) and the weighted sum of RMSE and Binary Cross Entropy. Then, the sequence length of the LSTM is five in the proposed model. We select the weights with the smallest validation loss in 50 epochs.

The baseline model predicted the action class with a one-frame image, whereas the proposed model uses five-frame images. Table 2 shows the accuracies with a 0.5 threshold. Fig. 13 shows the outputs of each recovery classifier on the test data in $\mathcal{D}_{\text{rec}}$. The proposed model was superior to the baseline model. In particular, the proposed model could predict the "recovery class" for wire deformations in data ID 271 to 287. The accuracy declines in the proposed model because of mistakes around the data ID 325. Test data around ID 325 showed that the wire deformed despite successful insertion. Therefore, the recovery classifier of the proposed model mistakes by recognizing the deformations of the wire.

Table 2. Accuracy of Test Data

Policy Model	Accuracy %
Baseline Model[15]	82.6
Proposed Model	85.5

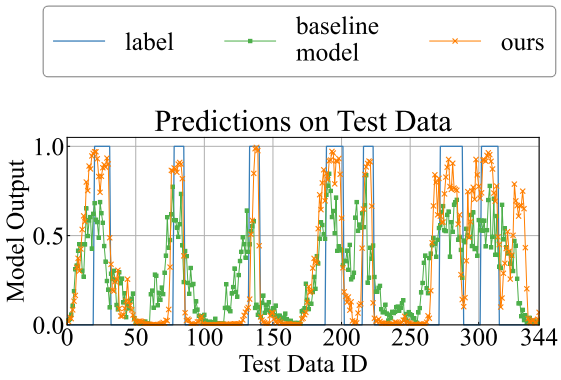


Fig. 13. Prediction Results of Action Labels. 5 demonstrations are consisted of 344 pairs of consecutive evaluation data and inputted into each model in order. The unmarked line shows the correct label, and the vertical value 1.0 indicates the "recovery class," whereas 0.0 indicates the "insertion class." The square line shows the predictions of the recovery classifier of the baseline model. Moreover, the crosses-line shows the predictions of the proposed model. The data numbers 271 to 287 indicate human intervention in the deformation of the wire. While the other cases where the correct labels are 1.0 indicate human intervention for the failure that the wire passed through the outside of the hole.

4.5 Wire Insertion

We conduct experiments with real-space wire insertions of the model having sub-modules. We initialize the model using weights in sections 4.3 and 4.4. The threshold of the recovery classifier is 0.6, 0.7, and 0.8, and Table 3 shows the best result.

In the baseline model, the wire passed outside the hole nine times, and the sub-modules pulled back the wire seven times to avoid insertion errors. However, the recovery classifier could not select the recovery action in the remaining two cases. Other failures included two failures in which the robot could not recover to the normal state and one failure in which the wire bent.

In the proposed model, there was one failure, the wire passed through the outside of the hole. We confirmed the input data in this failure; the wire was unseen without enough light reflection. In other cases, the wire collided and bent four times. The proposed model could pull back and insert the wire successfully, as shown in the case of Fig. 14. However, as we pointed out in the experiment in Section 4.4, the recovery classifier of the proposed model overreacted to the deformation of the wire, and a total of eight times incorrectly selected recovery action amid the appropriate insertion.

Table 3. Success Rate with Sub-modules

Policy Model (Sub-modules Available)	Success Rate %	Number of Successes times
Baseline Model [15]	86.1	31/36
Proposed Model	97.2	35/36

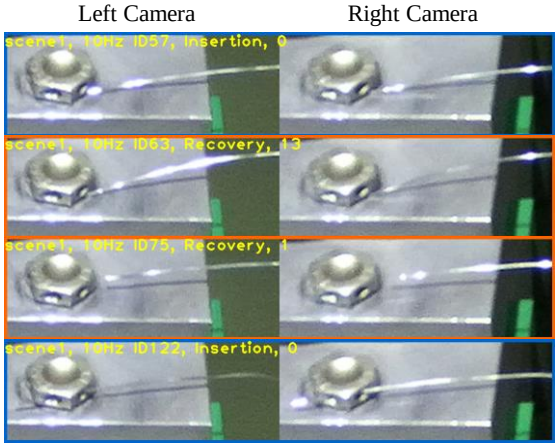


Fig. 14. Success with Pull-back Action

4.6 Wire Insertion (No Blackout Curtains + Automatic Picking)

In experiments 4.3 to 4.5, we used blackout curtains and one fixed light. In addition, we manually attached the wire to the robot's fingers. Finally, this paper performs insertion tests in the experimental setup that assumes real-world use. The new robot system does not have blackout curtains. It is a more complex environment because other lights affect the stereo camera. Moreover, the system can pick up the wire through pre-determined motion (Fig 15), and there is also randomness in the grasping position.

First, this section conducts insertion tests using the proposed model without sub-modules. We train the CNN[2] and the insertion module using $\mathcal{D}_{\text{ins}}$ (600 insertions, 65,135 training data, 5,556 validation data, and 1,350 test data) and Adam (learning rate 10^{-4}). The first row of Table. 4 shows a decline in the success rate. The proposed model tended to fail to cope with more intricate changes in the wire posture. In particular, the proposed model failed when the input was an image, in which the wire tip was challenging to see. Therefore, we improved the proposed model by concatenating no-wire images ($\mathbf{L}^{-1}$, $\mathbf{R}^{-1}$) into current images ($\mathbf{L}^{t}$, $\mathbf{R}^{t}$), as shown in Fig. 16. As a result, the success rate improved, as shown in Table 4, and the wire insertions were successful even for images where the wire was difficult to see.

Finally, the proposed model with sub-modules and no-wire image inserts the wire. We train the sub-modules using $\mathcal{D}_{\text{rec}}$ (300 insertions, 28,494 training data, 2,868 validation data, and 785 test data), sequence length 10, Adam (learning rate 10^{-4}), and the classifier threshold 0.5. The success rate was 96.3% (104/108), and in three cases, the insertion was successful by pulling back the wire with the recovery action, as shown in Fig. 17 (a) and (b). In four failures, the wire movement stopped halfway and could not entirely pass through the hole. We expect this case can improve by removing small actions from the insertion data.

Table 4. Success Rate (No Blackout Curtains + Automatic Picking)

Policy Model	Test Time (PM) & Weather	Success Rate %	Number of Successes
Proposed Model (No Sub-modules)	1:00-3:00 (sunny)	52.8	19/36
	1:00-3:00 (cloudy)	61.1	22/36
	6:00-9:00 (night)	50.0	18/36
Proposed Model + ($\mathbf{L}^{-1}$,$\mathbf{R}^{-1}$) (No Sub-modules)	1:00-3:00 (sunny)	80.6	29/36
	1:00-3:00 (cloudy)	86.1	31/36
	6:00-9:00 (night)	91.6	33/36
Proposed Model + ($\mathbf{L}^{-1}$, $\mathbf{R}^{-1}$) (Sub-modules Available)	1:00-3:00 (sunny)	97.2	35/36
	1:00-3:00 (cloudy)	97.2	35/36
	6:00-9:00 (night)	94.4	34/36

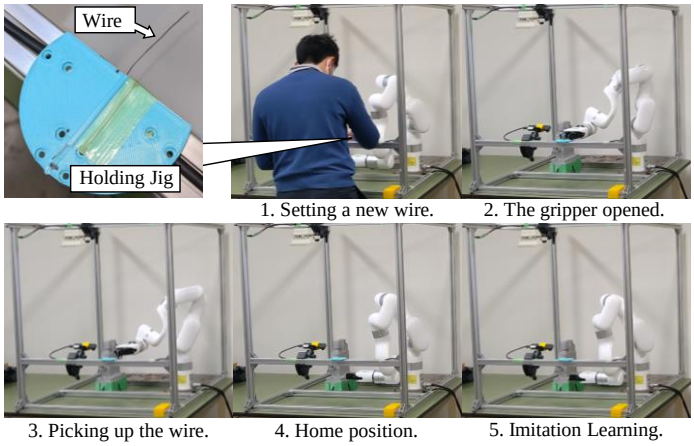


Fig. 15. Automatic Picking using Pre-determined Motion

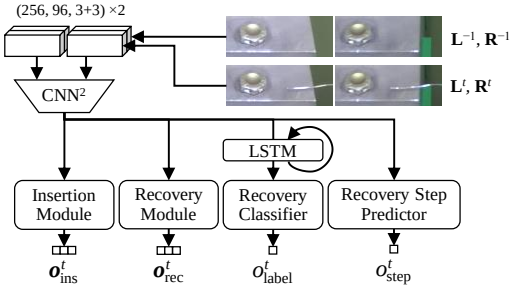
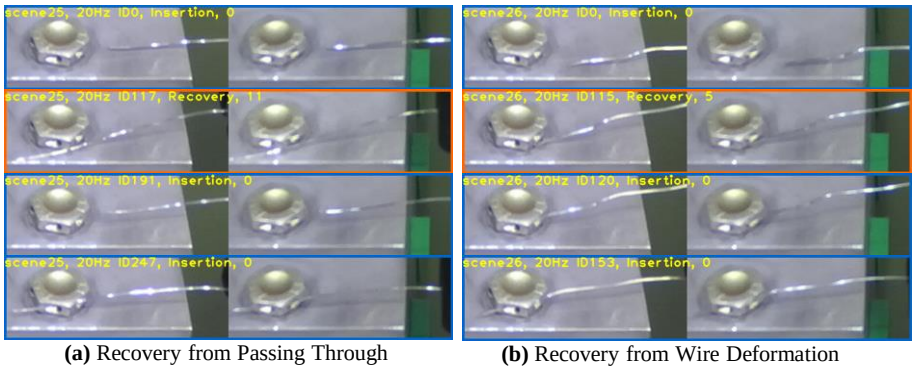


Fig. 16. Proposed Model + No-wire Images



(a) Recovery from Passing Through **(b)** Recovery from Wire Deformation

Fig. 17. Success Trials with Recovery Action

5 Conclusion

This paper automated the wire insertion task with uncertain component position and posture by imitation learning. We introduced CNN^2, LSTM, and no-wire images for

the baseline model. Then we proposed a novel data collection approach LHI to omit manual labeling and to collect the recovery action. In addition, we extend the triangular noise to three dimensions to reduce the covariate shift. We conducted the real-space wire insertion tests. The results showed that the proposed model could insert the wire with a higher success rate and recover from failures even if the wire deformed. In the last experiment, we evaluated our approach with no blackout curtains and automatic grasping and improved the success rate by inputting no-wire images.

The future issue is expanding the random range. The proposed model in this paper only controls the linear velocity of the robot tip. Therefore, it is necessary to improve the model and the teleoperation system by adding rotation commands.

References

1. N. Hogan: Impedance control: An approach to manipulation: Part II-Implementation. In: Journal of Dynamic Systems, Measurement, and Control, Vol. 107, No. 1, pp. 8-16 (1985)
2. A. Hussein, M. M. Gabar, E. Elyan and C. Jayne: Imitation Learning: A Survey of Learning Methods. In: ACM Computing Surveys (CSUR), Vol.50.2, No.21, pp.1-35 (2017)
3. M. Nigro, M. Sileo, F. Pierri, K. Genovese, D. D. Bloisi, Caccavale and F. Caccavale: Peg-in-Hole Using 3D Workpiece Reconstruction and CNN-based Hole Detection. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4235-4240 (2020)
4. S. R. Chhatpar and M. S. Branicky: Search strategies for peg-in-hole assemblies with position uncertainty. In: 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems, Expanding the Societal Role of Robotics in the Next Millennium, pp. 1465-1470 (2001)
5. J. C. Triyonoputro, W. Wan and K. Harada: Quickly inserting pegs into uncertain holes using multi-view images and deep network trained on synthetic data. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5792-5799 (2019)
6. J. Redmon and A. Farhadi: YOLOv3: An incremental improvement. In: arXiv:1804.02767 (2018)
7. P. Cirillo, G. Laudante and S. Pirozzi: Vision-Based Robotic Solution for Wire Insertion with an Assigned Label Orientation. In: IEEE Access, Vol. 9, pp. 102278-102289 (2021)
8. D. De Gregorio, R. Zanelli, G. Palli, S. Pirozzi and C. Melchiorri: Integration of Robotic Vision and Tactile Sensing for Wire-Terminal Insertion Tasks. In: IEEE Transactions on Automation Science and Engineering, Vol. 16.2, pp. 585-598 (2018)
9. G. Palli, and S. Pirozzi: A Tactile-Based Wire Manipulation System for Manufacturing Applications. In: Robotics, Vol. 8.2, No. 46 (2019)
10. S. Levine, C. Finn, T. Darrell and P. Abbeel: End-to-End Training of Deep Visuomotor Policies. In: The Journal of Machine Learning Research, Vol. 17.1, pp. 1334-1373 (2016)
11. T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg and P. Abbeel: Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 5628-5635 (2018)
12. T. Yu, C. Finn, A. Xia, S. Dasani, T. Zhang, P. Abbeel and S. Levine: One-Shot Imitation from Observing Humans via Domain-Adaptive Meta-Learning, In: arXiv:1802.01557 (2018)

13. J. S. Dirtside, E. R. Aye, A. Stahl and J. R. Matthiessen: Teaching a Robot to Grasp Real Fish by Imitation Learning from a Human Supervisor in Virtual Reality. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 7185-7192 (2018)
14. H. Kim, Y. Ohmura and Y. Kuniyoshi: Using human gaze to improve robustness against irrelevant objects in robot manipulation tasks. In: IEEE Robotics and Automation Letters, Vol. 5.3, pp. 4415-4422 (2020)
15. H. Kim, Y. Ohmura and Y. Kuniyoshi: Gaze-based dual resolution deep imitation learning for high-precision dexterous robot manipulation. In: IEEE Robotics and Automation Letters, Vol. 6.2, pp. 1630-1637 (2021)
16. H. Kim, Y. Ohmura and Y. Kuniyoshi: Memory-based gaze prediction in deep imitation learning for robot manipulation. In: arXiv:2202.04877 (2022)
17. H. Kim, Y. Ohmura and Y. Kuniyoshi: Transformer-based deep imitation learning for dual-arm robot manipulation. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 8965-8972 (2022)
18. H. Kim, Y. Ohmura and Y. Kuniyoshi: Robot peels banana with goal-conditioned dual-action deep imitation learning. In: arXiv:2203.09749 (2022)
19. A. Sasagawa, K. Fujimoto, S. Sakaino and T. Tsuji: Imitation Learning Based on Bilateral Control for Human-Robot Cooperation. In: IEEE Robotics and Automation Letters, Vol. 5.4, pp. 6169-6176 (2020)
20. S. Sho: Bilateral Control-Based Imitation Learning for Velocity-Controlled Robot. In: 2021 IEEE 30th International Symposium on Industrial Electronics (ISIE). pp. 1-6 (2021)
21. S. Sakaino, K. Fujimoto, Y. Saigusa and T. Tsuji: Imitation Learning for Variable Speed Object Manipulation, In: arXiv:2102.10283 (2021)
22. T. Kitamura, S. Sakaino, M. Hara and T. Tsuji: Bilateral Control of Human Upper Limbs Using Functional Electrical Stimulation Based on Dynamic Model Approximation. In: IEEJ Journal of Industry Applications, 20009551 (2021)
23. K. Hayashi, S. Sakaino and T. Tsuji: An Independently Learnable Hierarchical Model for Bilateral Control-Based Imitation Learning Applications. In: IEEE Access, Vol. 10 pp. 32766-32781 (2022)
24. Y. Saigusa, S. Sakaino, T. Tsuji: Imitation Learning for Nonprehensile Manipulation through Self-Supervised Learning Considering Motion Speed. In: IEEE Access, Vol. 10, pp. 68291-68306 (2022)
25. H. Kim, Y. Ohmura, A. Naga Kubo and Y. Kuniyoshi: Training Robots without Robots: Deep Imitation Learning for Master-to-Robot Policy Transfer. In: arXiv:2202.09574 (2022)
26. S. Ross, G. J. Gordon and J. A. Bagnell: A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. In: arXiv:1011.0686 (2010)
27. M. Laskey, J. Lee, R. Fox, A. Dragan and K. Goldberg: DART: Noise Injection for Robust Imitation Learning. In: Conference on robot learning, PMLR, pp. 143-156 (2017)
28. L. Ke, J. Wang, T. Bhattacharjee, B. Boots and S. Srinivasa: Grasping with Chopsticks: Combating Covariate Shift in Model-free Imitation Learning for Fine Manipulation. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 6185-6191 (2021)
29. F. Codevilla, M. Muller, A. Lopez, V. Koltun and A. Dosovitskiy: End-to-end Driving via Conditional Imitation Learning. In: 2018 IEEE International Conference on Robotics and Automation (ICRA) (2018)
30. W. Chen and S. Wu: CNN^2: Viewpoint Generalization via a Binocular Vision. In: Neural IPS 2019, pp. 1986-1998 (2019)
31. S. Hochreiter and S. Jürgen: Long short-term memory. In: Neural Computation 9(8), pp. 1735-1780 (1997)