2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

CFVS: Coarse-to-Fine Visual Servoing for 6-DoF Object-Agnostic Peg-In-Hole Assembly

Bo-Siang Lu[1], Tung-I Chen[1], Hsin-Ying Lee[1], and Winston H. Hsu[1,2]

***Abstract*— Robotic peg-in-hole assembly remains a challenging task due to its high accuracy demand. Previous work tends to simplify the problem by restricting the degree of freedom of the end-effector, or limiting the distance between the target and the initial pose position, which prevents them from being deployed in real-world manufacturing. Thus, we present a Coarse-to-Fine Visual Servoing (CFVS) peg-in-hole method, achieving 6-DoF end-effector motion control based on 3D visual feedback. CFVS can handle arbitrary tilt angles and large initial alignment errors through a fast pose estimation before refinement. Furthermore, by introducing a confidence map to ignore the irrelevant contour of objects, CFVS is robust against noise and can deal with various targets beyond training data. Extensive experiments show CFVS outperforms state-of-the-art methods and obtains 100%, 91%, and 82% average success rates in 3-DoF, 4-DoF, and 6-DoF peg-in-hole, respectively.**

I. Introduction

With the trend of industrial automation, intelligent robotic manipulation systems are expected to replace manual work and facilitate precision manufacturing. Despite the rapid advances in this area, peg-in-hole assembly remains a challenging task due to its low tolerance for deviations: a slight error on keypoint estimation could cause assembly failures. Though the task has been extensively studied in previous literature, developing a peg-in-hole approach with high accuracy and dexterity toward real-world manufacturing remains an open problem.

Prior work tends to simplify the problem by considering only limited degrees of freedom (DoF) (such as 3-DoF [1]–[4] and 4-DoF [5], [6]) or poses strict distance constraints between the targets and end-effector [7]–[9]. However, under such simplified settings, these methods could completely fail when dealing with more complicated pose differences between the targets and the end-effector, such as rotations and large initial alignment errors (see Tab. II). Specifically, prior methods can hardly handle rotational deviations, even with just a slight tilt angle, for the insertion direction is assumed to be constantly aligned with the z-axis of the hole during training, Moreover, since the methods based on force-torque feedback control need physical contact, and the visual servoing methods suffer from error accumulation, none of the existing methods can effectively handle large initial alignment errors, and thus could fail when the end-effector is far from the targets.

To address these issues, we present a novel Coarse-to-Fine Visual Servoing (CFVS) 6-DoF peg-in-hole approach that can tackle large initial alignment errors and perform 6-DoF motion control by leveraging 1) 3D point-cloud information and 2) a coarse-to-fine offset prediction pipeline. The proposed CVFS consists of two major components, the open-loop control with object-agnostic keypoint network (OAKN) and the visual servoing with offsets prediction network (OPN). The OAKN estimates the rough target position and guides the end-effector towards the objective. The OPN then gradually aligns the peg to the hole according to the received 3D visual feedback. In comparison to pure closed-loop control methods, this coarse-to-fine strategy alleviates large initial alignment errors by rapidly decreasing the range of exploration, avoiding error accumulation caused by step-by-step refinement.

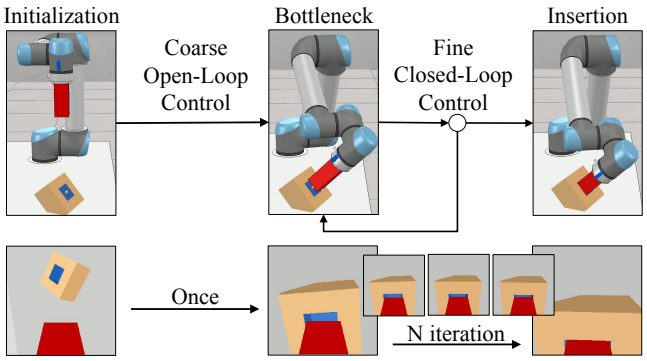

Fig. 1. **An overview of CFVS for 6-DoF peg-in-hole.** Integrating the concepts of open-loop control and closed-loop control, we propose CFVS which consists of the coarse and fine approaches. First, we directly move the end-effector to the bottleneck pose, overcoming the challenge of the large initial error. Then, we align the peg to the hole gradually with visual feedback, achieving a high insertion success rate in 6-DoF peg-in-hole.

Instead of taking 2D images [2], [4], [5], CFVS operates with 3D point-cloud inputs, which offer depth information of input objects and enable 6-DoF movement control. CFVS can therefore achieve 6-DoF insertion by encoding the 3D spatial relationship between pegs and targets. In addition, by adopting a confidence map that concentrates on the information of holes and disregards that of object contours, CVFS captures the local perception of target objects, which makes the insertion robust and agnostic to the variation of object shapes.

We outperform all state-of-the-art approaches and obtain 100%, 91%, and 82% average success rates in 3-DoF, 4-DoF, and 6-DoF peg-in-hole with 30 cm initial error. Experiment results demonstrate that CFVS succeeds in tackling large initial errors and diverse shape variations. Ablations (Sec. IV-F) also highlight the importance of our coarse-to-fine strategy.

[1]National Taiwan University, [2]Mobile Drive Technology

In brief, our main contributions are as follows:

- We propose CFVS, a coarse-to-fine 3D point-based visual servoing framework, which is the first to achieve 6-DoF peg-in-hole assembly with tilted holes.
- CFVS can achieve accurate insertion even with large initial alignment errors.
- CFVS is robust to various shapes of target objects and can generalize to unseen objects.

II. Related Work

A. Peg-In-Hole Assembly Task

Many researchers focus on the peg-in-hole problem which has no general formulation to solve [10]. Some works [1], [7]–[9], [11]–[18] regard peg-in-hole assembly as a contact-rich problem. They use the force-torque sensor to search a hole so that they can achieve high accuracy. However, in their method, the end-effector needs physical contact with target objects.

There exist some approaches using the visual sensor, as shown in Table I. For instance, M. Nigro et al. [19] leverage neural networks and traditional algorithms to detect the hole position and insertion direction. Although they can insert the peg into the tilted hole, they only use the open-loop method without further refinement. Compared to the method, the closed-loop approaches are more robust. J. C. Triyonoputro et al. [2] and R. L. Haugaard et al. [4] propose a learning-based visual servoing method to achieve 3-DoF peg-in-hole. Some methods [20], [21] use RL policy to accomplish the 3-DoF task. For 4-DoF insertion task, E. Valassakiand et al. [6] insert a square or cylindrical ring over a peg. They use Iterative Closest Point (ICP) [22] to approximately achieve a bottleneck pose followed by a closed-loop policy. E. Y. Puang et al. [5] use an autoencoder to learn self-supervised keypoint features and then train a visual servo network for insertion. Existing methods can not learn the 3D spatial relationship between the peg and hole and control the end-effector in 6-DoF. On the contrary, we utilize 3D information and are the first to achieve 6-DoF peg-in-hole with large initial alignment errors.

B. Coarse-to-Fine Strategy for Robotic Manipulation

The concept of coarse-to-fine strategy has been conducted for a long time [23], [24], which is a common method for robotic manipulation applications. Such a strategy can boost the performance and thus achieve a high success rate. E. Johns [25] uses an imitation learning method to learn for robotic manipulation. Their framework can be split into two phases: approach and interaction trajectory. E. Valassakis et al. [26] introduce a one-shot imitation learning method that learns from a demonstration. They move to a bottleneck pose based on visual servoing and then replay the demonstration. M. A. Lee et al. [20] use both model-based and model-free approaches. They estimate a rough pose to approach the object and then adopt a model-free policy for refinement. E. Valassaki et al. [6] propose a learning-based control policy. At first, they use Iterative Closest Point (ICP) [22] to estimate the object pose and then refine the pose to increase accuracy. Unlike these methods, which need to record the demonstration for each object in prior or can only operate with the same object, our coarse-to-fine pipeline can be agnostic to objects.

TABLE I: Comparison with vision-based approaches.

Method	6-DoF Peg-in-hole	Unseen Objects	Large Initial Alignment Error	Closed-loop Control
[2], [4], [5]	✗	✗	✗	✓
[6], [20], [21]	✗	✗	✓	✓
[19]	✗	✗	✓	✗
CFVS (Ours)	✓	✓	✓	✓

III. Method

We present CFVS, a coarse-to-fine 6-DoF peg-in-hole method comprised of two components: (A) the open-loop control with an object-agnostic keypoint network (OAKN) and (B) the visual servoing with an offsets prediction network (OPN), as shown in Fig. 2. The input 3D point clouds are reconstructed from depth images captured by the eye-in-hand camera in the world coordinate. We adopt PointNet++ [27] as the backbone network, which is a widely-adopted feature extractor for 3D robotic manipulation, to encode 3D points into high-dimensional features for further processing.

A. Object-Agnostic Keypoint Network (OAKN)

OAKN is designed to guide the end-effector towards the target position near the hole before further refinement. We propose to leverage point cloud features, which contain more abundant geometrical information than 2D images, to predict the required position and orientation information for open-loop control. The details are described as follows.

1) Open-Loop Control: Inspired by [28], the pose information for open-loop control can be represented by three keypoints, $K = \{k_1, k_2, k_3\}$. Specifically, k_1 denotes the central position $t \in \mathbb{R}^3$ of the hole, and the orientations along x- and z-axis can be computed by $v_x = k_2 - k_1$ and $v_z = k_3 - k_1$, respectively. The rotation matrix $R \in \mathbb{SO}^3$ can be computed according to these keypoints (see Algorithm 1), and the end-effector can be guided to the target pose $[R|t] \in \mathbb{SE}^3$ through inverse kinematics. The challenge lies in how to obtain high-quality keypoints based on unstructured 3D points captured by the depth camera. In the following sections, we will discuss how to generate candidate keypoints based on non-euclidean point cloud data and how to adaptively combine these candidates into the keypoints close to the target position.

2) Keypoint Prediction: Let $X = \{x_i\}_{i=1}^N$ be the input point cloud, where x_i is a point in 3D coordinates and N is the number of points. The backbone network encodes X into the feature $z \in \mathbb{R}^{N \times C}$, which will be sent to a Multi-Layer Perceptron (MLP) to predict the keypoint offsets $\{\Delta k_{i,1}, \Delta k_{i,2}, \Delta k_{i,3}\}$. Thus, through computing $k_{i,j} = x_i + \Delta k_{i,j}$, the N candidate keypoints K_i can be obtained:

$$K_i = \{k_{i,1}, k_{i,2}, k_{i,3}\}, \quad i \in \{1, 2, \ldots, N\}. \tag{1}$$

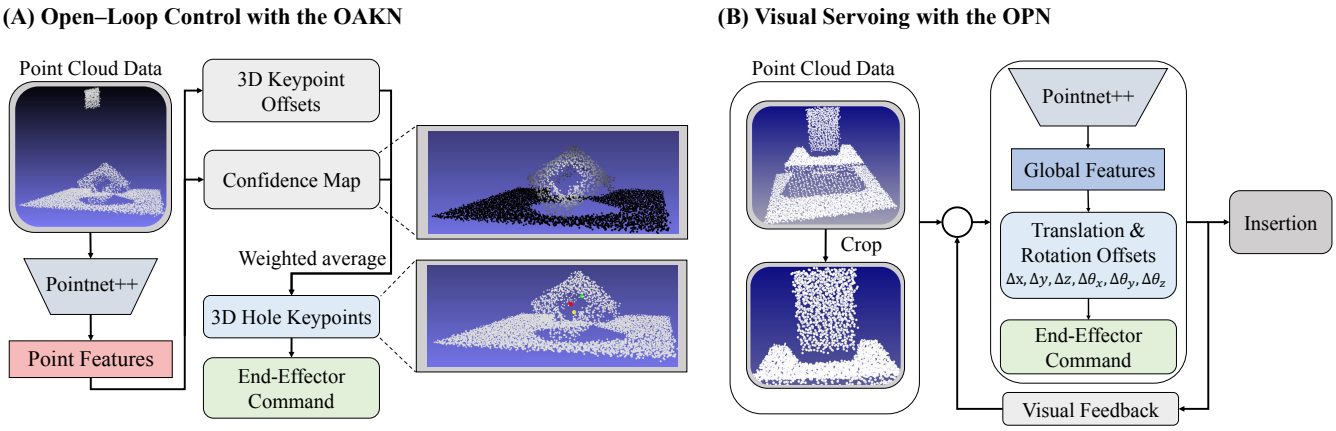

Fig. 2. **The 6-DoF peg-in-hole pipeline** contains two stages: (A) the open-loop control with the object-agnostic keypoint network (OAKN), which estimates the rough pose of a hole, and (B) visual servoing with the offset prediction network (OPN), which iteratively refines the final insertion position. In the first stage, OAKN takes a 3D point cloud as input and predicts candidate keypoints from each point. These keypoints, which represent the position and orientation of the hole, are weightedly averaged by the confidence heatmap that highlights points close to the hole and makes our pipeline object-agnostic. The end-effector then moves to that position and orientation. In the second stage, OPN extracts global features from the point cloud cropped around the hole and then generate the translation and rotation offsets for the end-effector according to the features. The OPN is iterated until we reach sufficiently small offsets. The insertion command is finally executed.

Naturally, one intuitive way to obtain the ultimate objective K for open-loop control is to average across all K_i. However, our experiment shows that such a naive average-pooling strategy leads to sub-optimal performance (as shown in Table IV). Therefore, we propose combining the candidate keypoints with a learnable confidence map that can adaptively re-weight the importance of K_i.

3) Confidence Map: After having multiple candidates, our next step is to determine the final keypoints to guide the end-effector towards the hole. However, since the central position of the hole is unknown at inference, we cannot directly assign the confidence score to each candidate keypoint according to the distance. Therefore, we introduce a learnable confidence map to re-weight each candidate keypoint and obtain the final keypoints. The confidence map is also predicted by a MLP, which receives z as the input and outputs $\{w_i\}_{i=1}^{N}$, $w_i \in [0, 1]$. Thus, the final keypoints K can be obtained by re-weighting each candidate keypoint:

$$k_j = \frac{\sum_{i=1}^{N} w_i k_{i,j}}{\sum_{i=1}^{N} w_i}, \quad j \in \{1, 2, 3\}. \tag{2}$$

The negative influence caused by irrelevant visual features, such as object contours and noisy backgrounds, can be eliminated through such an attention mechanism. As a result, the proposed model is robust against shape variation and can be applied to multiple different target objects (see Fig. 4).

4) Loss: OAKN is trained to minimize the loss $L_{\text{coarse}} = L_{\text{kpts}} + L_{\text{map}}$, where L_{kpts} is the loss of the predicted keypoint offsets and L_{map} represents the loss of the confidence map. Specifically, L_{kpts} is formulated as the weighted L1 loss between the predicted and ground truth keypoint offsets:

$$L_{\text{kpts}} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{3} w_i^* \| \Delta k_{i,j} - \Delta k_{i,j}^* \|, \tag{3}$$

Algorithm 1: Computation of rotation matrix

Input: 3D hole keypoints, k_1, k_2 and k_3.
Output: Rotation matrix, R.

Function `compute_rot_mat`(k_1, k_2, k_3)**:**
- $v_x \leftarrow k_2 - k_1$
- $v_z \leftarrow k_3 - k_1$
- $v_z^{norm} \leftarrow$ normalize_vector(v_z)
- $v_y \leftarrow$ cross_product(v_z, v_x)
- $v_y^{norm} \leftarrow$ normalize_vector(v_y)
- $v_x^{norm} \leftarrow$ cross_product(v_y, v_z)
- $R \leftarrow \begin{bmatrix} | & | & | \\ v_x^{norm} & v_y^{norm} & v_z^{norm} \\ | & | & | \end{bmatrix}$
- **return** R

where w_i^* denotes how important the candidate keypoints K_i are, and $\Delta k_{i,j}^*$ is the ground-truth 3D keypoint offset. Since the importance of K_i does not have ground truth, w_i^* is generated from a 3D Gaussian function:

$$w_i^* = e^{-\frac{1}{2\sigma^2}(\|x_i - p\|^2)}, \tag{4}$$

where $p \in \mathbb{R}^3$ is the central position of the hole and σ controls the range of the confidence map. Points closer to the position of the center are assigned higher confidence. This weighting mechanism makes OAKN ignore the contour of objects and backgrounds, paying more attention to the information near the hole center. On the other hand, L_{map} is formulated as the root mean square error, which forces the predicted confidence map to approximate a Gaussian distribution centered at the hole:

$$L_{\text{map}} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \|w_i - w_i^*\|^2}. \tag{5}$$

B. Visual Servoing with the Offsets Prediction Network

We adopt the visual servoing for further refinement in the second stage. OAKN predicts the pose of the end-effector close to the hole, but the pose is not accurate for insertion. Thus, we refine the pose via visual servoing with OPN, which estimates the translation and rotation offsets iteratively with visual feedback until we reach sufficiently small offsets or repeat up to the specific times.

1) Translation and Rotation Offsets: We refine the pose of the end-effector for precise insertion by estimating offsets that represent the pose difference between the current pose and the ideal target pose. We first crop the input point cloud $\{x_i\}_{i=1}^{N}$ into $\{x_i\}_{i=1}^{N'}$, where $N' \in [0, N]$ is the number of points near the end-effector. By cropping the point cloud, we focus more on the peg and hole and ignore other redundant information. Given the cropped point cloud, the OPN extracts the global features $\in \mathbb{R}^D$, which are fed into an MLP to predict the translation offsets $\Delta t = (\Delta x, \Delta y, \Delta z) \in \mathbb{R}^3$ in 3D coordinate and rotation offsets $\Delta r = (\Delta\theta_x, \Delta\theta_y, \Delta\theta_z) \in \mathbb{R}^3$ in Euler angle representation.

2) Visual Servoing: We use visual servoing, a closed-loop control method, to refine the pose of the end-effector for insertion by iteratively estimating the translation offsets Δt and rotation offsets Δr with the OPN. We first record the original pose of the end-effector $[R|t] \in \mathbb{SE}^3$. The rotation offset Δr is converted to the representation of rotation matrix $\Delta R \in \mathbb{SO}^3$. Then, we move the end-effector to the next pose $[R'|t'] \in \mathbb{SE}^3$, where $R' = \Delta R \cdot R$ and $t' = \Delta t + t$. Our fine approach, outlined in Algorithm 2, executes repeatedly until the predicted offsets are smaller than the error tolerance or the OPN is repeated up to specific times. Finally, we execute the insertion command to accomplish the task.

Algorithm 2: Visual servoing

Input: Point cloud, $\{x_i\}_{i=1}^{N'}$. Translation error tolerance, e_t. Rotation error tolerance, e_r.
Output: End-effector pose, ee.

while *True* **do**
  $\Delta t, \Delta\theta_r \leftarrow OPN(\{x_i\}_{i=1}^{N'})$
  $\Delta R \leftarrow convert\_to\_rotation\_matrix(\Delta\theta_r)$
  $R, t \leftarrow get\_ee\_pose()$
  $R' = \Delta R \cdot R$
  $t' = \Delta t + t$
  set ee to R', t'
  if $\Delta t < e_t$ *and* $\Delta\theta_r < e_r$ **then**
    break // servoing done
  end
end

3) Loss: We train the OPN with supervised learning by minimizing the loss $L_{\text{fine}} = L_{\text{trans}} + L_{\text{rot}}$, where L_{trans} and L_{rot} are the losses of translation and rotation offsets. For the translation offsets, we use both the root mean square error and cosine distance to optimize the translation error:

$$L_{\text{trans}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\|\Delta t - \Delta t^*\|^2} + (1 - \frac{\Delta t^T \Delta t^*}{|\Delta t||\Delta t^*|}), \quad (6)$$

where $\Delta t^* = (\Delta x^*, \Delta y^*, \Delta z^*)$ is the ground-truth translation offsets. We use the root mean square error to learn the magnitude of translation offsets and cosine distance to learn the direction of movement. For the rotation offsets, we adopt the root mean square error to minimize the rotation error:

$$L_{\text{rot}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\|\Delta r - \Delta r^*\|^2}, \quad (7)$$

where $\Delta r^* = (\Delta\theta_x^*, \Delta\theta_y^*, \Delta\theta_z^*)$ is the ground-truth rotation offsets.

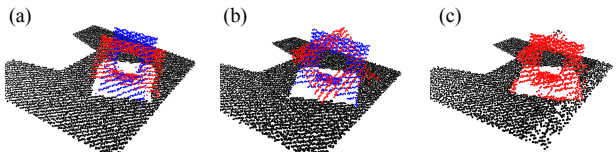

Fig. 3. **Data augmentation.** We apply three different augmentation: (a) random scaling, (b) random rotation and (c) mix of these two methods. (Best seen in color.)

C. Data generation

We collect coarse dataset D_{coarse} and fine dataset D_{fine} on Coppeliasim (formerly V-REP) [29]. The coarse dataset is trained on the OAKN while the fine dataset is trained on the OPN. For both the coarse and fine datasets, the hole object is randomly translated and rotated within the workspace.

1) Coarse Dataset: The coarse dataset is denoted as $D_{\text{coarse}} = \{X_i, \Delta K_i, W_i\}_{i=1}^{M}$, where M is the total number of the coarse data and X_i, ΔK_i and W_i are the point cloud, 3D keypoint offsets and confidence map under one scene. We collect the coarse dataset with M iterations. For each iteration, we keep the end-effector at an initial configuration and record the coarse data.

2) Fine Dataset: The fine dataset is defined as $D_{\text{fine}} = \{X_i, A_i\}_{i=1}^{L}$, where L is the total number of the fine data and X_i and A_i are the point cloud and actions under one scene. We collect the fine dataset with L iterations reversely. For every iteration, we set the pose of the end-effector to the ground-truth hole pose initially. We randomly decide the $-\Delta t$ and $-\Delta\theta_r$, which are the negative value of ground-truth translation and rotation offsets. Then, we move the end-effector according to the $-\Delta t$ and $-\Delta\theta_r$ and record the Δt and $\Delta\theta_r$ as fine data.

3) Data augmentation: Our training dataset is augmented with random scaling, random rotation, and a mix of these two methods, as shown in Fig. 3. We augment each object only along the x-axis and y-axis of the object frame. The points near the center of the holes are omitted to prevent the deformation of the holes. The augmentation increases the richness of the training set and boosts performance.

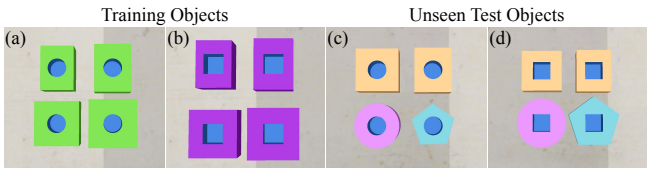

Fig. 4. **Train and Unseen Test Objects.** (a)(b) are training objects while (c)(d) are test objects. (a)(c) are round holes which are used for 3-DoF peg-in-hole, while (b)(d) are square holes which are used for 4-DoF and 6-DoF peg-in-hole.

IV. Experiments

We conduct a series of experiments to measure the performance of CFVS. We are curious about the accuracy, generalization capability, and efficiency. Thus, we want to examine: (1) How does our proposed method compare to other baselines in 3-DoF, 4-DoF, and 6-DoF peg-in-hole assembly with the small and large initial alignment error? (2) Can our proposed method generalize to unseen objects despite the large shape variation? (3) How fast is our overall framework with the small and large initial alignment error?

A. Experimental Settings

The experiments are running on Coppeliasim [29] with UR5 robot. We deploy our method on such an extendable simulation for a better comparison. We control the end-effector to the desired pose in 3D coordinate by inverse kinematics automatically solved by the simulator. In the simulated environment, the RGB-D camera is eye-in-hand, and the end-effector is a round or square peg strictly attached to the robotic arm. That is, there is no relative motion between the peg and the robotic arm. When testing, we add Gaussian noise $\mathcal{N}(0\text{ mm}, 1\text{ mm}^2)$ on a point cloud converted from a depth image. The radius and height of the round peg are 2.3 cm and 10 cm while the radius and depth of the round hole are 2.5 cm and 4.5 cm. The square peg is a 4.6 cm $\times$ 4.6 cm $\times$ 10 cm cuboid while the square hole is a 5 cm $\times$ 5 cm $\times$ 4.5 cm rectangular cavity. The clearance between the peg and hole is set to 4 mm. We set the initial distance between the peg and hole to 15 cm as the small initial alignment error and 30 cm as the large one.

B. Tasks

In our experiments, we design the 3-DoF, 4-DoF, and 6-DoF tasks. We test each object with 15 cm and 30 cm initial alignment error, respectively. All of these three tasks are with 4 mm clearance, the goal of the tasks is to insert a peg into a hole. For the shape of objects, we take four cuboids for training, and then use two similar unseen cuboids with small variation and two unseen shapes with large variation for testing, as shown in Fig. 4.

3-DoF Peg-in-hole. For 3-DoF, the peg and hole are round, and the object is randomly translated in XY-plane within the workspace. This is the simplest task because the end-effector can only move in the position of x, y, and z.

4-DoF Peg-in-hole. For 4-DoF, the peg and hole are square. The object can be randomly translated in XY-plane and rotated along the z-axis, which is perpendicular to the workspace plane. This task is more difficult because the end-effector needs to learn the movement of x, y, z, and yaw.

6-DoF Peg-in-hole. For 6-DoF, it is the most difficult of our tasks, and the peg and hole are square. The object can be randomly translated in XY-plane and rotated along the x-axis, y-axis, and z-axis. We set the tilt angle which is between the vertical axis and the direction of insertion to [0, 50] degrees due to the limitation of the end-effector. If the tilt angle is too large and the hole is back to the robotic arm, the end-effector can not reach such a pose. That is, there is no solution for inverse kinematics. The end-effector needs to learn the movement of x, y, z, roll, pitch, and yaw.

C. Evaluation Metrics

We use success rate to measure the performance. The depth of all the hole objects is 45mm. After the end-effector executes the insertion, if the peg touches the bottom of the hole, we regard this case as successful. Otherwise, it is a failure. Our proposed approach is compared to three different baselines. We test each unseen object 250 times, and the total is $250 \times 4 = 1000$ trials for each task. For the visual servoing methods, we additionally experiment with efficiency.

D. Baselines

ICP [22]. This approach, a conventional method for point cloud registration, is not a learning policy. We use the 3D model of 7 mm $\times$ 13 mm $\times$ 13 mm cuboid in the training set to estimate the object pose. The initial transformation matrix is set to the transformation which is from the position of the end-effector to the center point of the workspace.

ICP w/ kpts [22]. Similar to the ICP, ICP w/ kpts is additionally given a rough object pose. The pose is predicted by neural networks for the initial transformation matrix.

3DRHD [19]. This is an open-loop method estimating the hole position and insertion direction with an RGB-D camera. We implement the paper and use the point cloud which involves the whole scene instead of the object surface.

KOVIS [5]. We train KOVIS, a 2D visual servoing with the default settings. KOVIS is originally tested for their 4-DoF task with the round peg and hole. In our 4-DoF and 6-DoF tasks, we test KOVIS with the square peg and hole.

E. Results

Accuracy. We compare CFVS with four baselines. Table II depicts the success rate of peg-in-hole assembly with 4 mm clearance. For the 3-DoF task, ICP can hardly complete the insertion task without the accurate information of initial transformation. 3DRHD and ICP w/ kpts have lower performance because they are open-loop methods without further refinement. KOVIS and ours are competitive with 15 cm initial alignment error. However, KOVIS reduces the performance with 30 cm initial alignment error due to the pure visual servoing.

For the 4-DoF task, the success rates of all the baselines are drastically reduced. ICP w/ kpts can still insert into the

TABLE II: Success rate of peg-in-hole assembly with 4mm clearance. We test on 3-DoF, 4-DoF and 6-DoF peg-in-hole assembly with 15cm and 30cm initial alignment error. The result shows that ours outperforms to all other baselines with tilted holes and large initial alignment errors.

(a) 3-DoF peg-in-hole assembly (15 cm / 30 cm initial alignment error)

Method	Avg.	Small shape variation		Large shape variation	
		Cuboid_1	Cuboid_2	Cylinder	Pentagonal prism
ICP [22]	0.10/0.10	0.11/0.10	0.10/0.11	0.11/0.11	0.07/0.08
ICP w/ kpts [22]	0.67/0.69	0.71/0.74	0.77/0.76	0.69/0.70	0.52/0.54
3DRHD [19]	0.69/0.68	0.80/0.79	0.73/0.71	0.63/0.62	0.61/0.61
KOVIS [5]	0.95/0.80	**1.00**/0.88	**1.00**/0.87	0.86/0.71	0.94/0.75
CFVS (Ours)	**1.00/1.00**	**1.00/1.00**	**1.00/1.00**	**1.00/1.00**	**1.00/1.00**

(b) 4-DoF peg-in-hole assembly (15 cm / 30 cm initial alignment error)

Method	Avg.	Small shape variation		Large shape variation	
		Cuboid_1	Cuboid_2	Cylinder	Pentagonal prism
ICP [22]	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00
ICP w/ kpts [22]	0.23/0.22	0.55/0.56	0.26/0.25	0.01/0.00	0.08/0.07
3DRHD [19]	0.02/0.01	0.03/0.02	0.01/0.00	0.00/0.00	0.02/0.02
KOVIS [5]	0.16/0.02	0.18/0.03	0.26/0.03	0.06/0.01	0.12/0.02
CFVS (Ours)	**0.92/0.91**	**0.98/0.97**	**0.98/1.00**	**0.78/0.76**	**0.92/0.89**

(c) 6-DoF peg-in-hole assembly (15 cm / 30 cm initial alignment error)

Method	Avg.	Small shape variation		Large shape variation	
		Cuboid_1	Cuboid_2	Cylinder	Pentagonal prism
ICP [22]	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00
ICP w/ kpts [22]	0.21/0.20	0.64/0.63	0.08/0.07	0.05/0.03	0.05/0.06
3DRHD [19]	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00	0.00/0.00
KOVIS [5]	0.02/0.02	0.04/0.03	0.03/0.03	0.01/0.00	0.00/0.00
CFVS (Ours)	**0.82/0.82**	**0.92/0.93**	**0.90/0.94**	**0.76/0.72**	**0.70/0.69**

hole with small shape variation because the target object is similar to the given 3D model. However, they suffer from the large shape variation. KOVIS fails with the square peg and hole. In their experiments, they only test with the round peg and hole, which can not prove that they learn the z-axis rotation.

For the 6-DoF task, ours has the best performance, showing that CFVS is robust to tilted holes by comprehending the 3D relationship of peg and hole. Moreover, the experiments show that CFVS can be agnostic to objects, solving the problem of large shape variation.

TABLE III: Visual servoing time in second. We compare ours with KOVIS [5]. Although KOVIS [5] is faster with 15 cm initial alignment error, they reduce the performance with 30 cm initial alignment error. We keep the same speed with both 15 cm and 30 cm initial alignment errors.

Method	Initial alignment error	
	15 cm	30 cm
KOVIS [5]	**6.6**	10.1
CFVS (Ours)	7.0	**7.1**

Efficiency. In terms of the visual servoing approach, we compare CFVS with KOVIS for the efficiency with 15 cm and 30 cm initial alignment error. For the sake of fairness, we record the visual servoing time in 3-DoF peg-in-hole, as shown in Table III. We observe that CFVS takes about 7 seconds to complete the task with both 15 cm and 30 cm initial alignment error while KOVIS tends to spend more time with 30 cm initial alignment error. This is because KOVIS needs to move step by step. On the contrary, CFVS achieves the approximate pose at first and then refines the pose to complete the task.

F. Ablations

Table. IV demonstrates the result of the ablations. We carry out the ablation study in 6-DoF peg-in-hole with four unseen objects. Our key component is the coarse-to-fine (C2F) framework which boosts the performance a lot. Without refinement, we can not insert into the hole successfully due to the requirement of small error tolerance. Besides, the confidence map (Map) can help the model generalize to unseen objects with the large shape variation. Without the confidence map, we just calculate the average across all K_i and the insertion success rate will decrease. Moreover, data augmentation (Aug) is also a useful strategy that increases the richness of our training set. Without the data augmentation, the performance will be slightly influenced.

TABLE IV: The ablation study. (b,e) shows that the coarse-to-fine approach is crucial to our overall framework. (c,e) shows that using confidence confidence map improves the performance. (d,e) shows that using data augmentation can increase the success rate.

	C2F	Map	Aug	Avg.
a				22%
b		✓	✓	51%
c	✓		✓	73%
d	✓	✓		77%
e	✓	✓	✓	82%

V. Conclusion

In this paper, we propose CFVS, a coarse-to-fine 3D point-based visual servoing framework, which is the first to achieve the 6-DoF peg-in-hole assembly with tilted holes. CFVS alleviates the issue of large initial error in common visual servoing approaches with a fast and rough pose estimation before gradual refinement. This method greatly reduces the exploration range. In addition, by paying attention to only the information around holes, CFVS generalizes well to unseen objects and is robust to the variation of target shapes. However, although we succeed in 6-DoF, there is still room for improvement. In our 6-DoF experiments, we only test on the square hole and 4 mm clearance. In the future, we can extend our framework to adapt to more different shapes of holes and insert with a tighter clearance.

VI. Acknowledgement

This work was supported in part by National Science and Technology Council, Taiwan, under Grant NSTC 111-2634-F-002-022, and Mobile Drive Technology Co., Ltd (MobileDrive). We are grateful to the National Center for High-performance Computing.

REFERENCES

[1] J. Luo, E. Solowjow, C. Wen, J. A. Ojea, and A. M. Agogino, "Deep reinforcement learning for robotic assembly of mixed deformable and rigid objects," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2062–2069.

[2] 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)*. IEEE, 2019, pp. 5792–5799.

[3] M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg, "Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8943–8950.

[4] R. L. Haugaard, J. Langaa, C. Sloth, and A. G. Buch, "Fast robust peg-in-hole insertion with continuous visual servoing," *arXiv preprint arXiv:2011.06399*, 2020.

[5] E. Y. Puang, K. P. Tee, and W. Jing, "Kovis: Keypoint-based visual servoing with zero-shot sim-to-real transfer for robotics manipulation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 7527–7533.

[6] E. Valassakis, N. Di Palo, and E. Johns, "Coarse-to-fine for sim-to-real: Sub-millimetre precision across wide task spaces," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5989–5996.

[7] L. Johannsmeier, M. Gerchow, and S. Haddadin, "A framework for robot manipulation: Skill formalism, meta learning and adaptive control," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 5844–5850.

[8] P. Zou, Q. Zhu, J. Wu, and R. Xiong, "Learning-based optimization algorithms combining force control strategies for peg-in-hole assembly," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 7403–7410.

[9] S. Jin, X. Zhu, C. Wang, and M. Tomizuka, "Contact pose identification for peg-in-hole assembly under uncertainties," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 48–53.

[10] 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.

[11] Y. Fei and X. Zhao, "An assembly process modeling and analysis for robotic multiple peg-in-hole," *Journal of Intelligent and Robotic Systems*, vol. 36, no. 2, pp. 175–189, 2003.

[12] Y.-L. Kim, H.-C. Song, and J.-B. Song, "Hole detection algorithm for chamferless square peg-in-hole based on shape recognition using f/t sensor," *International journal of precision engineering and manufacturing*, vol. 15, no. 3, pp. 425–432, 2014.

[13] T. Tang, H.-C. Lin, Y. Zhao, W. Chen, and M. Tomizuka, "Autonomous alignment of peg and hole by force/torque measurement for robotic assembly," in *2016 IEEE international conference on automation science and engineering (CASE)*. IEEE, 2016, pp. 162–167.

[14] T. Tang, H.-C. Lin, Y. Zhao, Y. Fan, W. Chen, and M. Tomizuka, "Teach industrial robots peg-hole-insertion by human demonstration," in *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 2016, pp. 488–494.

[15] T. Inoue, G. De Magistris, A. Munawar, T. Yokoya, and R. Tachibana, "Deep reinforcement learning for high precision assembly tasks," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 819–825.

[16] K. Van Wyk, M. Culleton, J. Falco, and K. Kelly, "Comparative peg-in-hole testing of a force-based manipulation controlled robotic hand," *IEEE Transactions on Robotics*, vol. 34, no. 2, pp. 542–549, 2018.

[17] Z. Liu, L. Song, Z. Hou, K. Chen, S. Liu, and J. Xu, "Screw insertion method in peg-in-hole assembly for axial friction reduction," *IEEE Access*, vol. 7, pp. 148 313–148 325, 2019.

[18] 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," *Applied Sciences*, vol. 10, no. 19, p. 6923, 2020.

[19] M. Nigro, M. Sileo, F. Pierri, K. Genovese, D. D. Bloisi, 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)*. IEEE, 2020, pp. 4235–4240.

[20] M. A. Lee, C. Florensa, J. Tremblay, N. Ratliff, A. Garg, F. Ramos, and D. Fox, "Guided uncertainty-aware policy optimization: Combining learning and model-based strategies for sample-efficient policy learning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 7505–7512.

[21] D. Bogunowicz, A. Rybnikov, K. Vendidandi, and F. Chervinskii, "Sim2real for peg-hole insertion with eye-in-hand camera," *arXiv preprint arXiv:2005.14401*, 2020.

[22] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. Spie, 1992, pp. 586–606.

[23] S. Salcudean and C. An, "On the control of redundant coarse-fine manipulators," in *1989 IEEE International Conference on Robotics and Automation*. IEEE Computer Society, 1989, pp. 1834–1835.

[24] A. Sharon and D. Hardt, "Enhancement of robot accuracy using endpoint feedback and a macro-micro manipulator system," in *1984 American Control Conference*. IEEE, 1984, pp. 1836–1845.

[25] E. Johns, "Coarse-to-fine imitation learning: Robot manipulation from a single demonstration," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 4613–4619.

[26] E. Valassakis, G. Papagiannis, N. Di Palo, and E. Johns, "Demonstrate once, imitate immediately (dome): Learning visual servoing for one-shot imitation learning," *arXiv preprint arXiv:2204.02863*, 2022.

[27] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in neural information processing systems*, vol. 30, 2017.

[28] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, and J. Sun, "Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 632–11 641.

[29] E. Rohmer, S. P. Singh, and M. Freese, "V-rep: A versatile and scalable robot simulation framework," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1321–1326.