2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Scalable Network and Adaptive Refinement Module for 6D Pose Estimation of Diverse Industrial Components*

Kun Qian, Mustafa Suphi Erden, and Xianwen Kong[1]

***Abstract*— The estimation of the 6D pose of industrial components is essential for smart manufacturing. Especially for complex units that require intensive manual operations, such as a concentrator photovoltaics solar panel, accurate spatial localization provides visual aids for industrial automation. In this paper, we propose an accurate and scalable framework to address the dimensional variability of industrial components and tackle practical implementation issues. First, we use the scalable architecture EfficientNet as the backbone coupled with an enhanced feature pyramid network to estimate the object's pose. By introducing vertical and horizontal connections of shallow layers, the feature extraction of small objects is optimized for better detection accuracy. Second, leveraging the reliable 2D detection results and geometry information, an adaptive pose refinement module is designed to adjust the estimated 6D pose. The scaling of the backbone network and the computational complexity of refined modules are uniformly adjusted via a shared hyperparameter, resulting in a globally scalable framework. In terms of the pose estimation accuracy, the effectiveness of the refinement module and the real-time performance, validations are conducted both on the LINEMOD dataset and our customized datasets comprising of objects from the industrial photovoltaic system. Additionally, to further illustrate the effectiveness of the proposed method, a precision parallel robot is employed to validate the accuracy of real-time object pose tracking.**

I. INTRODUCTION

Recent developments in intelligent manufacturing and human-robot collaboration require more accurate localization and pose recognition of industrial components [1]. Computer vision (CV) methods associated with deep learning techniques are extensively investigated to accomplish such tasks, i.e., object 6D pose estimation [2]–[4]. However, the unique characteristics of industrial components and the practical application scenarios raise new problems that need prompt solutions. For instance, handling texture-less and metallic components, dealing with objects of diverse sizes and balancing the trade-off between accuracy and real-time performance.

6D pose estimation is a highly integrated CV task, including a series of subtasks, e.g., target recognition [5], semantic segmentation [6], and 3D transformation parameterizations [7]. Therefore, mature deep learning frameworks are adopted to first obtain 2D detections. Subsequently, solving separate Perspective-n-Point (PnP) problems for objects by their 2D-3D keypoints information [8] or dense 2D-3D correpondences [3]. More intuitively, some end-to-end pose regression convolutional neural networks (CNN) [2], [4], [9] are proposed. By deftly incorporating pose-related features with existing backbone network, i.e., VGG [10], YOLO [11] and Efficientnet [6], one generalized model can resolve multiple objects and the computational complexity is object-independent. However, current methods focus on improving the overall estimation accuracy and ignore the size diversity in practical datasets, typical with industrial components in automatic assembly tasks. As shown in Fig. 1, the state-of-the-art approaches achieve excellent overall results but a performance degradation is obvious for objects with smaller sizes.

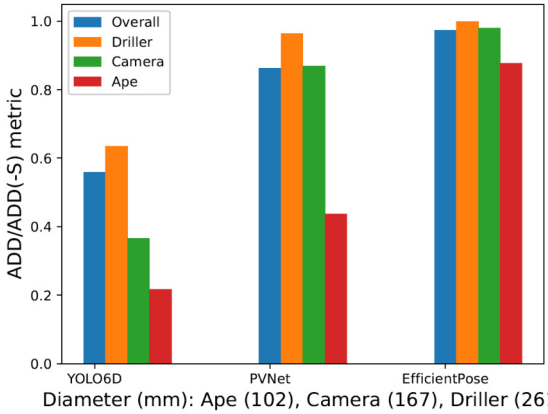


Fig. 1: The performance of different 6D estimation networks [4], [8], [9] on LINEMOD [12] datasets. Three typical object sizes and pose estimation accuracy are shown with ADD metrics.

Compared to the instance segmentation task with annotated 2D bounding boxes, the pixel-wise features extracted by CNN do not have rich spatial relationships for object 6D pose estimation [13]. Therefore, once the initial pose is obtained by the end-to-end network, extra refinement modules are designed with iterative closest points (ICP) [2], CNN [14] and recurrent neural networks [15]. Although accuracy is improved, the additional computational burden limits its real-time performance. In addition, the characteristics of these powerful backbone networks are not effectively leveraged, e.g. for accurate segmentation ability [16], mask prediction capability [17] and scalability [18].

In this work, we propose an accurate and scalable object pose estimation framework to address the aforementioned

*This research was fully funded by the Engineering and Physical Sciences Research Council (EPSRC) of the UK, with the Grant Reference EP/T024844/1.

[1] Corresponding author

K. Qian is with the School of Engineering, University of Manchester. (email: kun.qian@manchester.ac.uk)

M. Erden and X. Kong are with the School of Engineering and Physical Sciences, Heriot-Watt University. (email: m.s.erden@hw.ac.uk and x.kong@hw.ac.uk)

limitations. The scalable architecture Efficientnet [4], [18] is employed and the feature pyramid network (FPN) is enriched with shallow layer connections. Thus, relevant features of small objects that are easily lost in convolutional processing are preserved for better detection results. Subsequently, reliable 2D segmentation is used to direct a geometry-guided point cloud cropping. The computational complexity and point cloud registration accuracy are adaptively adjusted through the hyperparameters inherited from the Efficientnet backbone. Experiments are conducted on public datasets and on our customized datasets containing small (minimum dimensions are $18\,\mathrm{mm}\times14\,\mathrm{mm}\times5\,\mathrm{mm}$) industrial components from a photovoltaic solar unit. Validations on the trained networks, refined modules and robot-aided pose tracking illustrate the efficacy of the proposed method. The contributions of this paper are as follows

- The FPN is enriched with shallow layer connections along both fusion directions, with the aim of better preserving the estimation accuracy of small objects.
- A pose refinement module based on adaptive point cloud registration is proposed, facilitating the construction of the globally scalable 6D pose estimation framework with a shared hyperparameter.
- Datasets containing typical industrial components are established. The proposed method is validated on both publicly available and customized datasets. Additionally, practical implementation and real-time relative pose tracking are presented using an industrial parallel robot.

II. RELATED WORKS

A. 6D Pose Estimation

The two-stage 6D pose estimation approach, i.e., 2D detection with PnP solution [3], [8], [9], [19], [20] can be further categorized into two main methodologies: keypoint-based and intensive 2D-3D correspondence. Keypoint-based methods involve predicting the positions of eight 2D projections of the object's 3D cuboid as key points or selecting keypoints on the object's surface, where the algorithms like farthest point sampling are commonly used. Since key points may become occluded or truncated in the image, PVNet [8] employs a pixel-wise voting scheme, with each pixel of the object predicting a vector field pointing towards a keypoint. On the other hand, the dense 2D-3D correspondence methods aim to predict the 3D model points corresponding to each 2D pixel of the object. These dense 2D-3D correspondences can be established using UV maps [19] or transformed into coordinates within the 3D model space of the object [3]. Subsequently, the random sample consensus (RANSAC) iteratively optimizes this hypothetical set which leads to a universal n-PnP solution.

These studies report commendable computational speeds and accurate estimation results. Nevertheless, owing to their indirect pose regression natural, the quantity of objects significantly influences the real-time performance. Consequently, we opt for an end-to-end approach [4] to facilitate the customization of industrial applications.

B. Pose Refinement Methods

Regressing the 6D pose of objects in a single shot using CNN is a challenging task. Therefore, many existing researches concentrate on refining the 6D pose obtained from the initial estimation. This refinement process primarily relies on optimized point cloud registration algorithms [2], [9] or additional neural network architectures. Leading-edge studies [15], [21], [22] use the current estimate of object pose to render the 3D model, then adopt optical flow networks to regress the pose residual. DeepIM [21] and CosyPose [22] are built upon this concept, and the latter one improves the network architecture and rotation parameterizations. Furthermore, the introduction of geometric constraints and the utilization of Gated Recurrent Units have been proposed in [15] to facilitate a bi-directional dynamic pose refinement. Given an accurate and robust 2D detection, our insight is to fully utilize the available geometric information to adaptively adjust the complexity of the refinement algorithm.

C. 2D Detection and Backbone Network

For different detection scenarios, e.g., satellite images, numerous research endeavors have been directed towards addressing inconsistencies in object sizes within the context of 2D detection [23]. This issue is also common in industrial scenarios, notably in precision assembly. However, there is a lack of studies that consider the performance of 6D pose estimation for objects with diverse sizes. Furthermore, considering different usages of the CV system, which encompass one-shot localization, interval-based inspections, and real-time pose tracking, it becomes imperative to consider the scalability [4], [6], [18] of the pose estimation frameworks. Hence, this paper aims to develop an enhanced feature fusion network to improve the detection accuracy of small objects, while incorporating an adaptive refinement module to achieve the scalability of the overall framework.

III. APPROACH

In this paper, we first estimate the 6D pose of industrial components in RGB images by an end-to-end network. Subsequently, with the depth information, 2D segmentations are leveraged to facilitate an adaptive pose refinement module. To enhance the estimation accuracy of small objects, shallow layer features are incorporated in the multi-scale feature fusion. Moreover, the backbone network [6] employs a compound scaling method that jointly scales up all dimensions of the backbone with a single hyperparameter. Thus, the refinement module inherits the hyperparameter to achieve a trade-off between computational complexity and accuracy.

A. Pose Estimation Network and Feature Fusion

We use the EfficientNet [6] as the backbone and intuitively expand the network for object pose estimation. EfficientNet employs a compound scaling method for object detection, which jointly scales up the resolution/depth/width for all backbone, feature network and the prediction networks (object class, bounding box and etc.) with a hyperparameter ϕ (range from 0 to 7), as shown in Fig. 2(a). Since Efficientnet

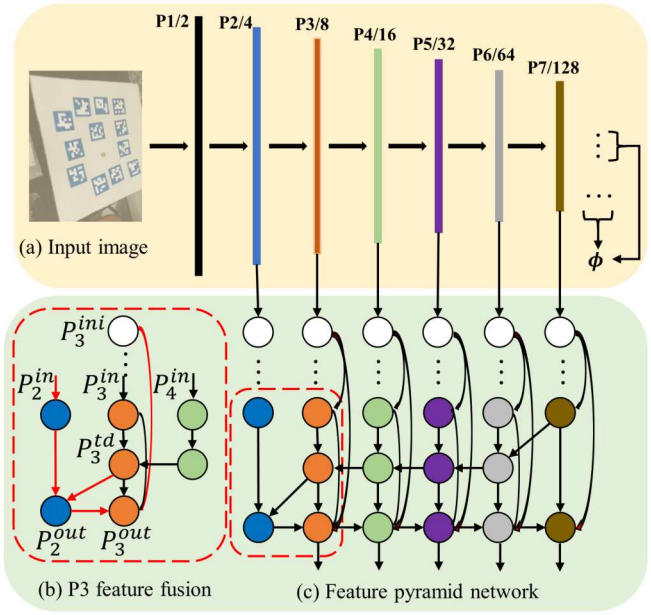


Fig. 2: Structure schematic diagram of the network. White and colored circles are initial feature inputs and features of different levels. ϕ is the compound scaling factor and $\vdots$ implies repeated depthwise separable convolution blocks. (a) Input image and the backbone network that are jointly scalable by hyperparameter ϕ, (b) Level three feature fusion (P_3) with shallow layer connection (red arrows and curve), (c) The feature pyramid contains top-down pathway, bottom-up pathway and direct shallow layer connections.

is an anchor-based method, four subnetworks are used to predict the object class, 2D bounding box, rotation R and translation t of each anchor box.

To preserve the accuracy of detection of small objects, for which the area of the object bounding box is less than 32×32 pixels, we enrich the multi-scale feature fusion in [18] with additional shallow layer connections, as shown in Fig. 2(b) and (c). Firstly, in a vertical way, we include the second input feature level in the fusion process. The shallow layer contains rich spatial information that directly use it to enter the final regression, classification and pose estimation detector can improve the small object detection accuracy. Meanwhile, aiming at the problem that features of small objects are easily lost in the convolution process, we add the direct connection layer to concatenate the initial input feature map with the output of each fusion iteration. We take the third layer P_3 to jointly explain the merges of upper and lower layers in both directions, shown by red arrows and curve in Fig. 2(b). Given the initial level three input feature P_3^{ini} and current iteration information: the input feature of level three P_3^{in} and level four P_4^{in}, the level two output feature P_2^{out}, the intermediate feature P_3^{td} and the output feature P_3^{out}, we have

$$P_3^{\text{td}} = \text{Conv}\left(\frac{\omega_1 \cdot P_3^{\text{in}} + \omega_2 \cdot \text{Resize}(P_4^{\text{in}})}{\omega_1 + \omega_2 + \epsilon}\right)$$
$$P_3^{\text{out}} = \text{Conv}\left(\frac{\omega_1' \cdot P_3^{\text{in}} + \omega_2' \cdot P_3^{\text{td}} + \omega_3 \cdot \text{Resize}(P_2^{\text{out}}) + \omega_4 \cdot P_3^{\text{ini}}}{\omega_1' + \omega_2' + \omega_3' + \omega_4 + \epsilon}\right) \quad (1)$$

where ω_i, ω_i' are learnable weights and ϵ is a small value to avoid numerical instability. Resize is an upsampling or downsampling operation for resolution matching. The rest of the connections are constructed in a similar manner that contain top-down pathway, bottom-top pathway and weighted optimization follow [18]. Notably, to further improve the efficiency, depthwise separable convolution [24] is used for feature fusion, as well as batch normalization and activation after each convolution. Moreover, the direct shallow connections are added in all layers for preserving spatial information, as illustrated in Fig. 2(c).

Three loss functions are employed as in [2], [4]. The first is the classification loss L_{class}, the second is the segmentation loss L_{box}, and the last is the overall pose loss L_{trans} (the same formula as ADD and ADD(-S) for asymmetry and symmetry objects in Section IV. A). To ensure equal contributions of different losses, we add weights γ_{class}, γ_{box} and γ_{trans} to each functions and the overall loss L is given by

$$L = \gamma_{\text{class}} \cdot L_{\text{class}} + \gamma_{\text{box}} \cdot L_{\text{box}} + \gamma_{\text{trans}} \cdot L_{\text{trans}}. \quad (2)$$

B. Rotation and Translation Representations

We use axis-angle representation to formulate the rotation of the object. Any rotation matrix R lies in the set of orthogonal matrices $SO(3) \doteq \{R : R \in \mathbb{R}^{3\times3}, R^T R = I_3, \det(R) = 1\}$. The geodesic distance $d(R_1, R_2)$ of two rotation matrices R_1, R_2, can be defined as [25]

$$d(R_1, R_2) = \frac{||\log(R_1 R_2^T)||_F}{\sqrt{2}} \quad (3)$$

where $\log(\cdot)$ is the matrix logarithm and $||\cdot||_F$ denotes the Frobenius norm. Each R captures the rotation of a 3D point by an angle $\theta \in \mathbb{R}$ about an axis $v \in \mathbb{R}^{3\times1}$, where $||v||_2 = 1$. By defining a skew-symmetric operator $[\cdot]_\times$, we have $R = \exp(\theta[v]_\times)$ where $\exp(\cdot)$ is the matrix exponential. Upon the unique correspondence between each axis-angle vector $r \in \mathbb{R}^3$ and the rotation matrix R, we further restrict that $\theta \in [0, \pi]$ and $R = I_3 \Leftrightarrow v = 0_3$. The matrix exponential can then be simplified to $R = I_3 + \sin\theta[v]_\times + (1 - \cos\theta)[v]_\times^2$ using the Rodrigues' rotation formula. Therefore, the axis-angle version of $d(\cdot, \cdot)$ can be expressed by [26]

$$d_{\text{aa}}(R_1, R_2) = \cos^{-1}\left[\frac{\text{tr}(R_1^T R_2) - 1}{2}\right] \quad (4)$$

where $\text{tr}(\cdot)$ denotes the matrix trace. For the translation vector $t = [t_x, t_y, t_z]^T$, we follow PoseCNN [2] that estimate the 3D translation by center point localization and object distance regression. With the 2D projection $c = (c_x, c_y)^T$ of t on the target object and depth information t_z, we can recover the t_x and t_y by assuming the following pinhole

PROCEDURE: Pose Refinement Module

Prepare: ϕ, R_{ini}, t_{ini}, W, H, W_{bb}, H_{bb}, $\text{Voxel}_{\text{ini}}$ and σ_{ini}.
Step 1: Segmentation-based point cloud cropping. (W_{bb}, H_{bb}, R_{ini} and t_{ini} used.)
Step 2: Constrained three-point pair sampling with fast point feature histogram (FPFH). (Geometry-aware point-point distance selection.)
Step 3: Calculating R_{opt} and t_{opt} with least square rigid transformation via singular value decomposition.
Step 4: Transformed point cloud validation within (ϕ+ $\text{Voxel}_{\text{ini}}$) interior threshold.
Step 5: Applying rejection ratio $(15 - 2\phi)\%$ (Matrix distance calculation.)
Step 6: Update pose residuals proportionally. (Normalization of $\text{Mean}(\frac{W_{\text{bb}}}{W} + \frac{H_{\text{bb}}}{H})$.)
Repeat Step 2 to **4** until $\sigma < \sigma_{\text{ini}}$ or reach a maximum of five iterations.

camera model:

$$\begin{bmatrix} t_x \\ t_y \end{bmatrix} = \begin{bmatrix} f_x \cdot \frac{t_x}{t_z} + p_x \\ f_y \cdot \frac{t_y}{t_z} + p_y \end{bmatrix} \tag{5}$$

where f_x and f_y are the focal lengths and $p = (p_x, p_y)$ denotes the principal point. Thus, we define c as the object 2D center and predict the offset in pixels for each anchor box. With predicted relative offsets, coordinate maps from the feature maps and the predefined strides, the absolute coordinate of c can then be calculated.

C. Adaptive RANSAC-like Scalable Refinement

In the EfficientPose [4], the rotation subnetwork predicts one r for each anchor box. Subsequently, an CNN-based refinement module is developed for iteratively updating a pose regression residual Δr. Although the number of convolution layers is uniformly regulated by the hyperparameter ϕ, without additional features such as optical flow (i.e., dense correspondence), the accuracy of such regression without a rendering operation is poor [3], [15], [27].

The performance of the Efficientnet has been verified on various datasets [6], [18], we decide to fully utilize its 2D detection capability. Therefore, we propose a geometry-guided procedure based on the available segmentation results to guide the refinement of the object pose. Based on 2D detection and depth information, the projected object point cloud is used as the source to match the cropped target point cloud. The compound scaling factor ϕ is inherited to achieve trade off between computational complexity and accuracy. The main concept is based on the fact that under the same camera perspective, the contours of small objects, especially the top vertices can be better captured. This provides a reference for finding the corresponding feature vector in point cloud registration. Although large objects tend to lose feature information under some relative pose, e.g., 180° reversal, the initial pose estimated by the neural network is with high accuracy. Thus, we design an adaptive rejection ratio and a fault tolerance range for the refinement module.

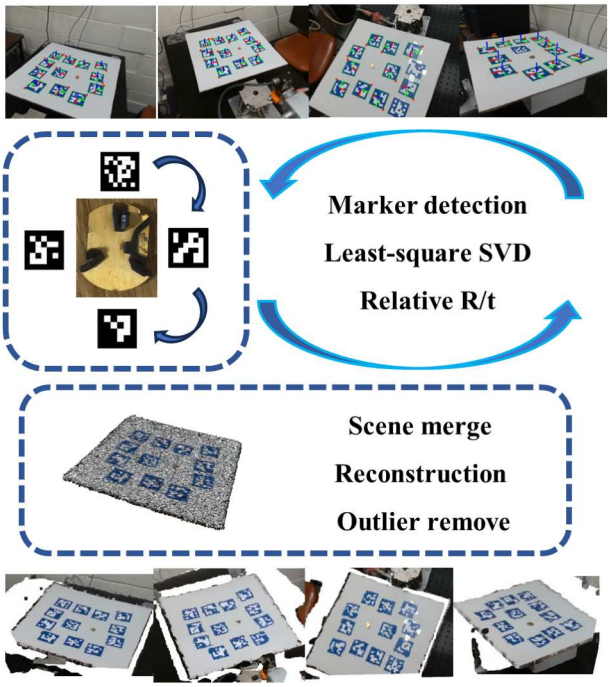


Fig. 3: The pipeline of dataset establishment. (a) Frame-wise transformation calculation via marker detection and least square singular value decomposition, (b) Obtaining the practical point cloud via scene combination, selective reconstruction and outlier removal.

The following parameters are introduced for establishing the refinement procedure:

- ϕ: A shared hyperparameter in initial pose estimation network.
- W, H, W_{bb} and H_{bb}: The width and height of input images and detected bounding boxes, respectively.
- R_{ini}, t_{ini}, R_{opt} and t_{opt}: The rotation matrix and translation vector before and after one refinement iteration.
- $\text{Voxel}_{\text{ini}}$, σ_{ini} and σ: The initial radius threshold for evaluating the point cloud registration, the transformation tolerance and the actual transformation difference calculated by $\sigma = |||[R_{\text{opt}}, t_{\text{opt}}] - [R_{\text{ini}}, t_{\text{ini}}]||_F$.

Note that the bounding box information W_{bb} and H_{bb} are obtained in parallel with the pose estimation results, thus will not bring additional computational burden. Furthermore, the refined pose will be validated after each iteration with a tolerance factor σ to decide on the termination of the iteration. The detailed procedures are summarized in the "PROCEDURE".

D. Custom Datasets Establishment

The objects we focus on are subparts of a solar photovoltaic unit. The dimensions of objects are as follows
1) Solar cell: metal color, size ($18\,\text{mm}$, $14\,\text{mm}$, $5\,\text{mm}$) and diameter ($23.35\,\text{mm}$)
2) Primary lens: transparency, size ($49\,\text{mm}$, $45\,\text{mm}$, $9\,\text{mm}$) and diameter ($67.13\,\text{mm}$).

Based on the definition of "small object" [23], the solar cell (area $< 32 \times 32$ pixels) can be classified as a small object, while the primary lens ($32 \times 32 <$ area $< 96 \times 96$ pixels) as a medium-sized object.

Our goal is to develop an intelligent and automated assembly system aided by the CV presented in this paper. Consequently, accurate 6D pose estimation of the involved components is one of the most important requirements and technology for this goal. To train the network, we develop the pipeline of dataset establishment, as shown in Fig. 3. The target object is surrounded by Aruco markers to obtain the ground truth pose. By slowly moving the camera (Intel RealSense depth camera D435i), the pose variations of the object relative to the camera are recorded. The 3D positions of marker corners are obtained using OpenCV library tools [28] and processed for depth information, and the rotation and translation matrices of two independent frames are calculated using singular value decomposition, intrinsic camera parameters and a threshold-based iteration. Regarding the 3D model of objects, through point cloud signal processing, the reconstructed 3D scene can be manually processed by MeshLab. After eliminating outliers, the axis-aligned bounding box of the object is placed in the center of camera perspective which can be used as an object model for template or mesh matching. Besides, applying computed rotations and translations on the segmented mesh with 3D to 2D projection also provides bounding box annotations. It is worth mentioning that this pipeline is also applicable for objects without precise 3D models.

IV. EXPERIMENTS

A. Datasets and Evaluation Metrics

The LINEMOD [12] dataset is used for validation, as well as the customized datasets introduced in Section III. D. In both LINEMOD and customized datasets, each object contains about 1200 RGB images. The training and test groups are randomly chosen and contain 15% and 85% images, respectively [8]. For asymmetric objects, we use the ADD metric [12] to evaluate the estimation accuracy. Given the ground truth R/t and the estimated one $\hat{R}/\hat{t}$, ADD computes the mean of pairwise distance between the 3D model transformed according to the ground truth and estimated pose, i.e.,

$$\text{ADD} = \frac{1}{m} \sum_{x \in \mathcal{M}} \left\| (Rx + t) - (\hat{R}x + \hat{t}) \right\|_2 \tag{6}$$

where m is the total number of 3D model points denoted as x. The set $\mathcal{M}$ contains all points x. For symmetric objects, ADD(-S) is then used by computing the closest distance between points $x_1, x_2 \in \mathcal{M}$:

$$\text{ADD(}-\text{S)} = \frac{1}{m} \sum_{x_1 \in \mathcal{M}} \min_{x_2 \in \mathcal{M}} \left\| (Rx_1 + t) - (\hat{R}x_2 + \hat{t}) \right\|_2 . \tag{7}$$

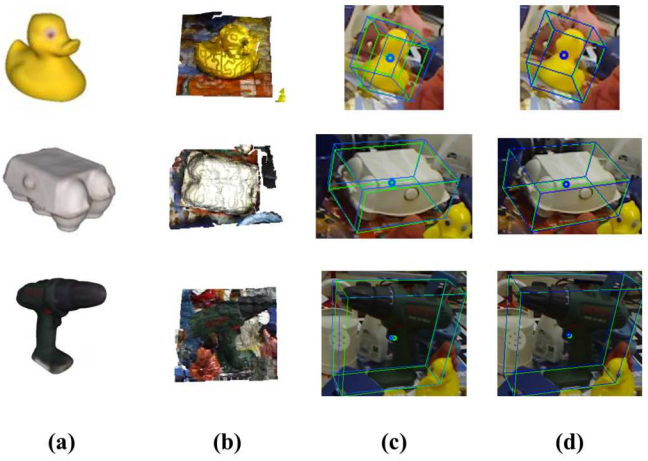


Fig. 4: Visualization of results on three objects in the LINEMOD datasets. Green bounding boxes represent the ground-truth poses while blue bounding boxes represent estimated results. (a) Object overview, (b) Segmented object point cloud, (c) Initial pose estimation, (d) The final pose with additional refinement module.

TABLE I: Object sizes, accuracy of [4], performance of refinement module, total number of refined images and count of total iterations.

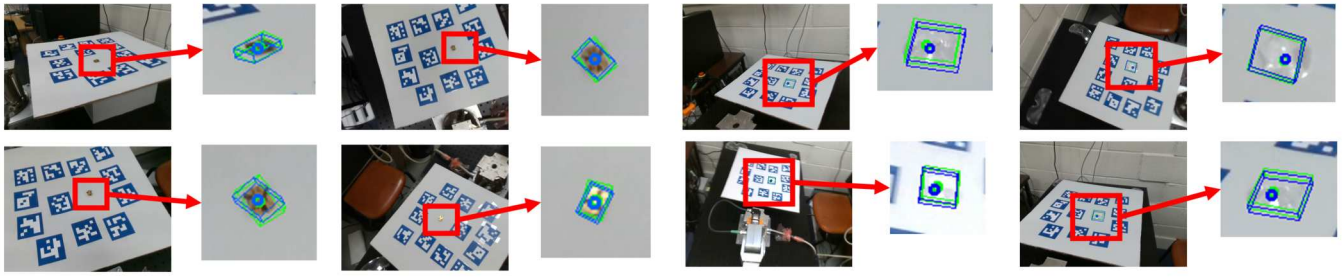
	Duck	Eggbox	Driller
Object size (diameter: mm)	109	164.63	261.47
ADD(-S, 1%) [4]	83.92	96.65	97.22
With refinement module	87.19	99.12	98.75
Refined image number	327	42	33
Iteration counts	1177	109	159

B. Implementation Details

We use Adam optimizer with an initial learning rate of 1e-4 and a minimal learning rate 1e-7. For the LINEMOD datasets, we set $\gamma_{\text{class}} = \gamma_{\text{box}} = 1$ and $\gamma_{\text{trans}} = 0.02$ in (2). For the customized datasets, since the solar cell and primary lens are extremely small and the magnitude of L_{trans} is determined by the object size, we set $\gamma_{\text{trans}} = 0.2$ to avoid domination of L_{class} and L_{box}. With the proposed FPN, we initialize Efficientnet with COCO pretrained weights. For the pose refinement module, the initial radius threshold and iteration termination tolerance are set as $\text{Voxel}_{\text{ini}} = 0.01$ and $\sigma_{\text{ini}} = 0.5$. In the experiment that involves the robot, Robot Operating System (ROS) is employed to achieve communication between the camera and the robot. With the Realsense SDK and Hexapod ROS driver, real-time images, robot control commands and real-time feedback can be accessed.

C. Refinement Results of LINEMOD Datasets

Three example objects [12] with obvious size differences are used, i.e., duck, eggbox and driller. We conduct evaluations of original EfficientPose [4] with $\phi = 0$ on the objects, which results in the performance shown in TABLE I. Note that, to further illustrate the performance of our method, we decrease the threshold of the ADD(-S) from 10% of object



(a) (b)

Fig. 5: Visualization of results on (a) solar cell datasets and (b) primary lens datasets with network scale $\phi = 0$.

diameter to 1%. By setting $\sigma = 0.5$, the accuracy with the refinement module, the number of refined images and the total refinement iterations are given. Fig. 4 visualizes the refinement process and all three examples perform one refinement iteration (the first iteration is not counted). Improvements of 13.27%, 2.47% and 1.53% are found for duck, eggbox and driller, respectively. It can be seen that an average refinement iteration 3.6, 2.6, and 4.81 are conducted on three objects. Theoretically, the average refinement iteration should increase as the object size increases, as smaller objects typically exhibit less self-occlusion and have fewer reconstructed point cloud data. The eggbox has fewer iterations mainly due to its flat surface.

TABLE II: Performance comparison of the proposed FPN and original EfficientPose (without refiner), proposed FPN with the refinement module and existing methods with refiners. Results are the average of the performance for the solar cell and primary lens datasets.

Method	ADD(-S)	5° 1 cm	Refine?
Proposed FPN	69.93	63.12	✗
EfficientPose [4]	65.63	59.17	✗
Proposed FPN + Refine	**83.97**	82.86	✓
CosyPose [22]	75.41	77.08	✓
CoupledRefine [15]	82.15	**84.19**	✓

5° 1 cm: rotation error is within 5° and the translation error is below 1 cm.

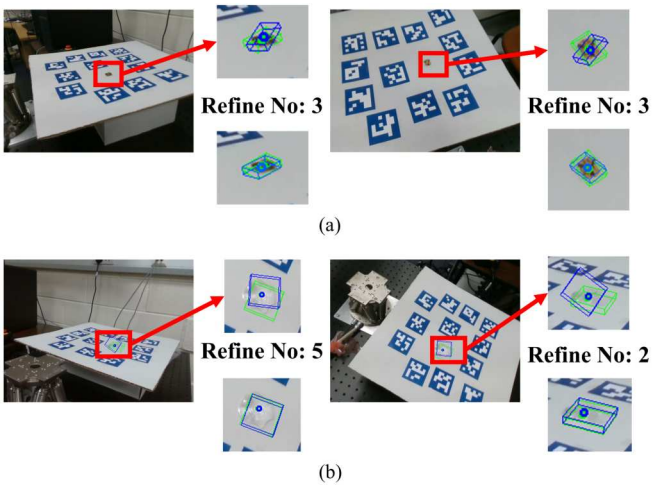


Fig. 6: Visualization of pose refinement results and iteration number on the (a) solar cell datasets and (b) primary lens datasets.

D. Custom Datasets Results

Performance of the Scalable Network. We first implement our enhanced feature fusion network and summarize the comparison of the results with [4], in TABLE II. Pose estimation results of customized datasets without activating the refinement module are illustrated in Fig. 5. It can be seen that, even without the refinement module, accurate poses can be predicted by the network. Moreover, shallow feature connections can effectively preserve the performance of small object detection. The final average ADD of the solar cell and primary lens dataset after 500 epochs are 69.93% and 65.63% for networks with and without shallow layer connections, respectively. Additionally, we demonstrate the performance of different methods using the 5° 1 cm metric, which measures the percentage of predicted rotations and translations that fall within 5° and 1 cm accuracy. An average improvement of 3.95% can be found after the shallow layer connections are applied. The size of the primary lens (49 mm×45 mm×9 mm) is approximately three times larger than the solar cell (18 mm×14 mm×5 mm), thus our proposed FPN demonstrates better performance, i.e., the improvements on the solar cell dataset (5.56% for ADD and 4.83% for 5° 1 cm) are more significant than on the primary lens dataset (2.43% for ADD and 3.07% for 5° 1 cm).

Effectiveness of the refinement module. To validate the effectiveness of the refinement module, we choose the network scale $\phi = 0$ for validation and implement two state-of-the-art 6D pose refinement frameworks [15], [22]. After each refinement process, we recalculate the mean ADD and 5° 1 cm metric and the statistical results are summarized in TABLE II. The visualization of four pose refinement procedures are given in Fig. 6, two examples for each of the solar cell and primary lens datasets. In TABLE II, it can be observed that the refinement module significantly enhances the ADD metric by 14.04% and outperforms geometric-based refinement [22] and the recurrent gate unit-based method in [15]. Although our proposed FPN with refinement module performs slightly worse than the CoupledRefine [15]

on the $5°$ $1\,\mathrm{cm}$ metric, our method is advantageous as it solely relies on geometry information (same as [22] and is 5.78% higher) and is training free. It's worth noting that the solar cell possesses rich surface information, thus leading to more stable performance. The transparency of the primary lens causes difficulties for image processing, as evident in Fig. 6(b) where a portion of the object remains uncovered in the ground truth pose annotation.

E. Real-time Performance Analysis

For practical implementation, we examine the real-time performance of our approach in several scenarios. The experiments are conducted with different network scales, i.e., $\phi = 0$ and $\phi = 3$. Also, the inference speed whether activating the pose refinement with the inherited hyperparameter is validated. All experiments are performed using a batch size of 1 and results are summarized in TABLE III. The following computational times are calculated

- preprocess the input image (Preprocessing),
- prediction time of the network (Network),
- the complete end-to-end time including the data preprocessing, network inference (non-maximum suppression), post-processing (rescaling) and pose refinement module (End-to-end).

All experiments are run on the same machine with an i7-6700HQ CPU and a 1080Ti GPU using Tensorflow 1.15.0. Based on the theoretical analysis, a larger ϕ will scale up the network, thus imposing a greater computational burden. As shown in TABLE III, when $\phi = 3$, the inference speed without activating the refinement module is merely 30% of that with $\phi = 0$. Apart from the influence of the hyperparameter setting, the real-time performance of the refinement module has to be considered. Given the substantial volume of point cloud data (15655 and 23471 points in total for the solar cells and primary lens, respectively), its real-time prediction speed is approximately $2\,\mathrm{fps}$. Notably, the refinement module with $\phi = 3$ is slightly faster than $\phi = 0$, attributed to the inclusion of adaptive thresholds. To conclude, the proposed network runs up to $16\,\mathrm{fps}$, which is sufficient for real-time pose estimation. In some scenarios with low operational speed, such as precision manufacturing and assembly, the proposed refinement module can further improve accuracy by appropriately sacrificing real-time performance.

F. Robot-driven Real-time Relative Pose Tracking

We use an industrial parallel robot, H-811.I2 Hexapod from Physik Instrumente® to validate the real-time pose tracking performance. The robot can achieve fast, stable and accurate 6-axis motion with a repeatability of $\pm 0.06\,\mathrm{mm}$. One translation and one rotation are chosen to validate the accuracy of the relative pose tracking. Initially, due to the absence of ground truth pose annotation in practice, the robot first moves to $-10\,\mathrm{mm}$ along its X-axis. Considering the current pose as the origin, the robot will then move to $10\,\mathrm{mm}$ and this $20\,\mathrm{mm}$ relative pose tracking results are compared with the actual robot feedback. Similarly, we employed the Y-axis for rotation with an initial value of $-10\,°$ and a total rotation of $20\,°$. The translation and rotation are conducted at the speed of $5\,\mathrm{mm/s}$ and angular speed of $230\,\mathrm{mrad/s}$, respectively.

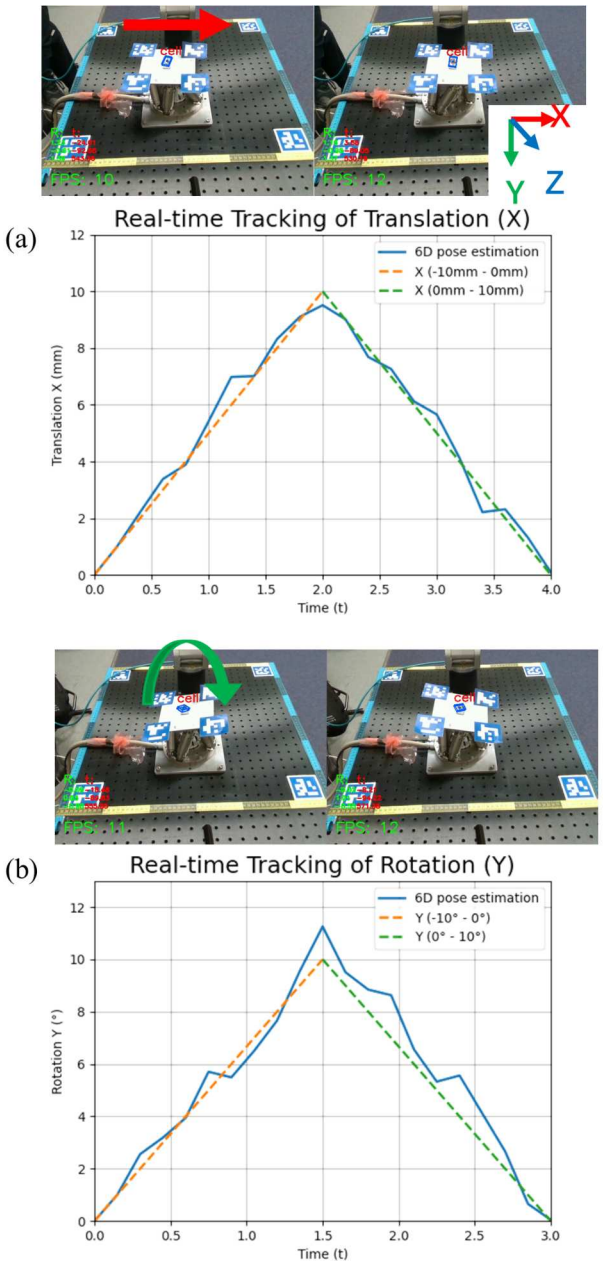


Fig. 7: Real-time pose tracking validations via the hexapod robot. (a) Translation along the X-axis and (b) Rotation along Y-axis.

Fig. 7 demonstrates the start and end of the hexapod robot motion. Moreover, we depict the image alongside the corresponding ground truth trajectory based on pose estimation sampling points. It can be seen that the algorithm can track translation and rotation well with a root-mean-square error and standard deviation of $1.14 \pm 0.81\,\mathrm{mm}$ and $1.61 \pm 1.31\,°$, respectively.

V. CONCLUSION

This paper studied the 6D pose estimation problem, addressing challenges posed by small object size and aiming

TABLE III: Real-time performance analysis of the pose estimation framework under different hyperparameters and with/without (w/o) the refinement module.

Hyperparameter (refinement)		$\phi = 0$ (w/o)	$\phi = 0$ (w)	$\phi = 3$ (w/o)	$\phi = 3$ (w)
Preprocessing	ms	14.63	14.63	45.5	45.5
	fps	68.35	68.35	21.98	21.98
Network	ms	48.49	48.49	149.27	149.27
	fps	20.62	20.62	6.7	6.7
End-to-end	ms	63.12	533.47 (470.35)	194.77	647.94 (453.17)
	fps	15.84	1.87 (2.13)	5.13	1.53 (2.21)

(*) specifies the time and fps of the refinement module.

at improving the scalability of the framework. Customized datasets were first established which contain two small industrial components. With the introduction of direct connections of shallow features during feature fusion, the pose estimation accuracy of industrial components can be improved by 5.56% compared to the exiting fusion network [18]. Building upon available 2D detections, an adaptable pose refinement module was introduced and can further enhance the accuracy by 15.29%. The framework was validated on the public datasets and its real-time performance was also demonstrated. Moreover, a hexapod robot was employed to validate the real-time pose tracking and the results further illustrated the efficiency of the proposed framework.

REFERENCES

[1] A. Shojaeinasab, T. Charter, M. Jalayer, M. Khadivi, O. Ogunfowora, N. Raiyani, M. Yaghoubi, and H. Najjaran, "Intelligent manufacturing execution systems: A systematic review," *Journal of Manufacturing Systems*, vol. 62, pp. 503–522, 2022.

[2] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *arXiv:1711.00199*, 2017.

[3] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6d object pose estimation by iterative dense fusion," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 3343–3352.

[4] Y. Bukschat and M. Vetter, "Efficientpose: An efficient, accurate and scalable end-to-end 6d multi object pose estimation approach," *arXiv preprint arXiv:2011.04307*, 2020.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[6] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, 2019, pp. 6105–6114.

[7] K. Kleeberger, M. Völk, R. Bormann, and M. F. Huber, "Investigations on output parameterizations of neural networks for single shot 6d object pose estimation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 13 916–13 922.

[8] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "Pvnet: Pixel-wise voting network for 6dof pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4561–4570.

[9] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 292–301.

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

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[12] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," in *Computer Vision–ACCV 2012: 11th Asian Conference on Computer Vision*, 2013, pp. 548–562.

[13] X. Jiang, D. Li, H. Chen, Y. Zheng, R. Zhao, and L. Wu, "Uni6d: A unified cnn framework without projection breakdown for 6d pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 174–11 184.

[14] S. Mahendran, H. Ali, and R. Vidal, "3D pose regression using convolutional neural networks," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 2174–2182.

[15] L. Lipson, Z. Teed, A. Goyal, and J. Deng, "Coupled iterative refinement for 6d multi-object pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6728–6737.

[16] B. Marin, K. Brown, and M. S. Erden, "Automated masonry crack detection with faster R-CNN," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, 2021, pp. 333–340.

[17] O. Skeik, M. S. Erden, and X. Kong, "6d pose estimation for precision assembly," in *2022 IEEE 5th International Conference on Image Processing Applications and Systems (IPAS)*, 2022, pp. 1–6.

[18] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 781–10 790.

[19] S. Zakharov, I. Shugurov, and S. Ilic, "Dpod: 6d pose object detector and refiner," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1941–1950.

[20] S. Lin, Z. Wang, Y. Ling, Y. Tao, and C. Yang, "E2EK: End-to-end regression network based on keypoint for 6d pose estimation," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6526–6533, 2022.

[21] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, "Deepim: Deep iterative matching for 6d pose estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 683–698.

[22] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "Cosypose: Consistent multi-view multi-object 6d pose estimation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*, 2020, pp. 574–591.

[23] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual generative adversarial networks for small object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1222–1230.

[24] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[25] C. Song, J. Song, and Q. Huang, "Hybridpose: 6d object pose estimation under hybrid representations," in *Proceedings of the IEEE/CVF Conference on Computer vision and Pattern Recognition, CVPR*, 2020, pp. 431–440.

[26] C. D. Meyer, *Matrix analysis and applied linear algebra*, 2023, vol. 188.

[27] A. Trabelsi, M. Chaabane, N. Blanchard, and R. Beveridge, "A pose proposal and refinement network for better 6d object pose estimation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 2382–2391.

[28] G. Bradski, "The opencv library." *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, vol. 25, no. 11, pp. 120–123, 2000.