

Received February 25, 2021, accepted March 16, 2021, date of publication March 24, 2021, date of current version April 7, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3068467

6D Pose Estimation of Occlusion-Free Objects for Robotic Bin-Picking Using PPF-MEAM With 2D Images (Occlusion-Free PPF-MEAM)

DIYI LIU^{1*}, SHOGO ARAI^{2*}, (Member, IEEE), YAJUN XU³,
FUYUKI TOKUDA², (Member, IEEE), AND KAZUHIRO KOSUGE², (Fellow, IEEE)

¹Mech-Mind Robotics Technologies, Beijing 100094, China

²Department of Robotics, Tohoku University, Sendai 980-8579, Japan

³Graduate School of Information Science and Technology, Hokkaido University, Sapporo 060-0814, Japan

Corresponding author: Shogo Arai (arai@tohoku.ac.jp)

*Diyi Liu and Shogo Arai are co-first authors.


ABSTRACT Pose estimation that locates objects in a bin is necessary for a robotic bin picking system. Although many algorithms have shown high performance in pose estimation, most algorithms estimate the poses of objects regardless of their occlusion. This can reduce the success rate in picking up the object. To resolve this issue, we propose a novel pipeline that estimates a pose only for occlusion-free objects based on point pair feature-based pose estimation with multiple edge appearance model (PPF-MEAM). The proposed method detects occlusion-free objects in the 2D image captured by a camera with a convolutional neural network framework. Next, corresponding point clouds of occlusion-free objects need to be extracted by using their locations in the 2D image. We propose a robust extraction method that finds the 3D points corresponding to image pixels in the 2D image to reduce the effect of the calibration errors between the camera and 3D sensor. The point cloud of the occlusion-free objects is finally input into a pipeline of PPF-MEAM to estimate the pose of the object. The experiment results prove that the proposed method is about 50% faster 30% higher in terms of pose estimation success rate compared with the original PPF. Moreover, it increases the success rate of picking tasks compared with the original PPF-MEAM.

INDEX TERMS 6D pose estimation, multiple edge appearance model (MEAM), occlusion-free, robotic bin picking.

I. INTRODUCTION

The last decade has seen a rapid increase in the introduction of industrial robots to production lines to increase their productivity and overcome labor shortages. However, some tasks conducted in an unstructured environment are still difficult to automate. A robotic bin-picking task is one such challenging task. To achieve the robotic bin-picking, the robot must locate an object placed among other objects in an unstructured environment, where the objects change their positions and orientations every time an object is removed from the bin. Locating a distinct object in a bin, in other words, pose estimation of a distinct object, has become possible with the introduction of commercialized three-dimensional (3D) sensors.

With rapid advances in 3D measurements [1], [2], such as stereo vision [3]–[5], active stereo methods [6], [7], time

The associate editor coordinating the review of this manuscript and approving it for publication was Guilin Yang .

of flight, a number of robotic tasks, such as positioning tasks [8], cooperative tasks [9]–[11], and bin-picking [12], [13], have been conducted using 3D sensors. This study focuses on pose estimation algorithms using 3D measurements. One of the major types of pose estimation algorithms utilizes descriptors to express the shapes of the target objects. With these algorithms, the target object is described using some descriptors offline. The descriptors are then used to find and estimate the poses of the objects in the scene. In general, pose estimation algorithms using only 3D data can be divided into two categories based on whether a global or local descriptor is applied.

Using global descriptors necessitates the segmentation of the individual object in the scene. The shape of the target object is described during the offline phase using global descriptors. Given the point cloud of a scene, the segmented point cloud of the object is used for computing the global descriptors. These computed global descriptors will be used for a comparison with the description, which is trained in

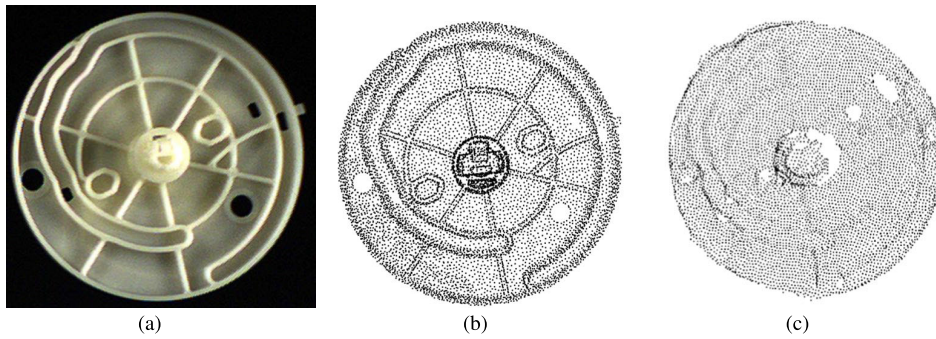


FIGURE 1. Defective point cloud of Part A. (a): photo of Part A. (b): model point cloud. (c): point cloud captured by the sensor. The detailed point cloud of the ridges on the part cannot be captured with the embedded 3D sensor algorithm. Thus, the appearance of the model point cloud differs from the data captured by the 3D sensor.

advance. The correspondence is then used to estimate the pose of the object. To shorten the time required for pose estimation, Aldoma *et al.* [14] proposed a descriptor called a clustered viewpoint feature histogram (CVFH) based on a smooth region growth algorithm. Harada *et al.* [15] applied a bin-picking task based on this method. Aldoma *et al.* then improved the CVFH using interpolation and called this new method an oriented, unique, and repeatable clustered viewpoint feature histogram (OUR-CVFH) [16]. However, these methods rely heavily on the results of the segmentation and have shown a limited performance for use with industrial parts.

Unlike global descriptors, local descriptors do not require segmentation [17]–[23]. Tombari *et al.* proposed a local 3D descriptor for surface matching, called a signature of histograms of orientations (SHOT) [19]. Because this descriptor's identity is extremely high, with a total of 352 dimensions (11×32), it achieves high-level performances in object recognition, 3D reconstruction, and shape retrieval. Although these descriptors have high discriminative power, the descriptors need to be computed for 3D keypoints and detection of 3D keypoints required a long time in a heavily cluttered scene. To shorten the time required for detection of 3D keypoints, Arai *et al.* [24] have proposed a fast detection method for 3D keypoints called FADA-3K. However, FADA-3K has not been applied to robotic bin-picking yet. Other local features [17]–[20] that use the geometric information around the key points also require a long time.

However, local descriptors, which utilize the relationship between multiple points [21]–[23] show a higher computational speed. A pose estimation algorithm using the point pair feature (PPF), proposed by Drost *et al.* [21], has attracted significant attention. In this method, the authors utilize both a PPF and a voting scheme within a Hough space, which realizes the pose estimation of industrial parts with the robustness of occlusion in a cluttered scene. As stated in [22], the performance of this method [21] deteriorates in the presence of numerous outliers and incorrect estimation of the normal.

Tuzel *et al.* [23] proposed a max-margin learning framework to identify discriminative features on the surface of the objects. For different tasks, the features are ranked according to their importance, which leads to improved accuracy and reduced computational cost. Kiforenko *et al.* [25] conducted a performance evaluation of PPFs. According to this study, a four-dimensional descriptor, i.e., a PPF, is the best choice for most datasets. In addition, it has a higher recall rate than other local histogram features.

Moreover, the aforementioned methods share a common problem, i.e., they are not robust to insufficient data captured using a 3D measurement sensor. As shown in Fig. 1, the appearance of the captured point cloud of Part A differs from the appearance of the model point cloud. Because this part is made from resin with 3-mm thick ridges on its surface, a point cloud captured around the ridges is significantly different compared with its appearance when applying currently existing 3D sensors. Because the captured point cloud does not have sufficient information near the ridges, the horizontal rotation of the pose of this part cannot be computed. To solve this problem, we proposed a point pair feature-based pose estimation with a multiple edge appearance model (PPF-MEAM) [33]. This method uses 3D point cloud data and 2D image data of the scene as the input; it has shown high performance in pose estimation experiments.

Nevertheless, after integrating it into a robotic bin picking system, we found that it cannot help the system achieve a high success rate because PPF-MEAM estimates the poses of the objects regardless of their occlusion, which will likely result in the robot picking an occluded object in the bin. Picking an occluded object increases the grasping error or failures in the completion of the picking-up task. If the robot can pick up an object under other objects, during the picking motion, the grasped object will make contact with other objects, as shown in Fig. 2. A collision results in a change in the pose of the object with respect to the gripper, likely resulting in a failure of the following placement task, particularly for heavy target objects. Even worse, the robot may fail to grasp the object completely because of the

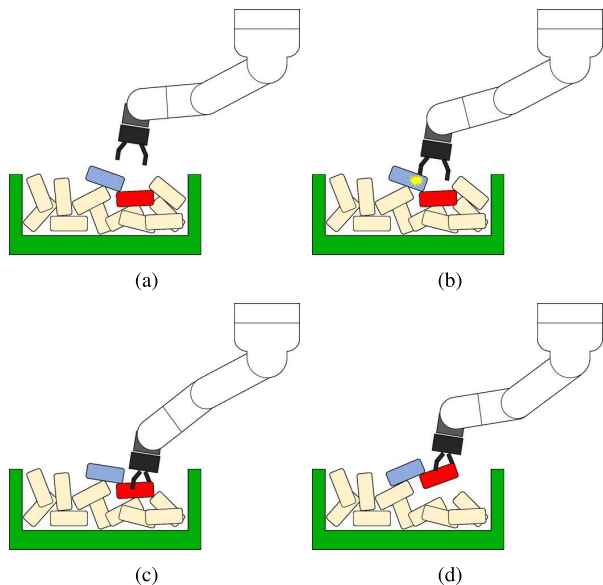


FIGURE 2. Cases of the failed picking task. (a) and (b) show that the robot fails to grasp the object up because there is no space for the gripper to grasp the occluded object. (c) and (d) show that the robot picks up the object under another object. During the picking motion, the grasped object contacts another object which results in grasping error.

likelihood that there will be no space for the gripper to grasp the occluded object, which is also a common issue for the aforementioned methods. To improve the performance of the robotic picking system, occlusion-free PPF-MEAM is proposed. We first find occlusion-free objects in a 2D image using a deep learning-based object detection method called RetinaNet [34]. At the same time, we apply the Canny edge detection for the 2D image to obtain the edge pixels. Using the results of the occlusion-free and edge detections, the edge pixels of an occlusion-free object can be obtained. We then extract 3D boundary points of occlusion-free objects in the point cloud using their corresponding 2D edge pixels. RetinaNet used here has also been modified and outputs not only the position of the objects but also the 2D feature points. In addition, 3D feature points, corresponding to the 2D feature points, will also be extracted for computing the PPF.

As another unique proposition, to compensate for the calibration error between the 2D camera and 3D sensor, a method for extracting 3D boundary points of occlusion-free objects in the point cloud using their corresponding 2D edge pixels is proposed. We call this method 3D boundary point exploration (3DBPE). To obtain the 2D image, the 3D point cloud, and the relationship between each pixel and each 3D point, we use a sensor set including a 2D camera and a 3D sensor. However, owing to the calibration error between the 2D camera and 3D sensor, we occasionally cannot find the corresponding points in the point cloud for the edge pixels. The 3DBPE approach is thus proposed to solve this problem. Finally, a modified PPF-MEAM line that uses the boundary-to-boundary-using-directional-tangent-line (B2B-DTL) PPF is utilized to estimate the pose of occlusion-free objects using their boundary

point cloud. To clarify the domain of the proposed method, we should point out that this method is designed for usual scenes in factories. In such scenes, we have only one category of parts that are randomly piled up.

The rest of this paper is organized as follows. As a foundation of this study, Section II provides an overview of the original PPF-MEAM. Section III introduces the details of the occlusion-free PPF-MEAM. In this section, the pipeline of the proposed method is first illustrated, and the method for detecting occlusion-free objects is then described. In the following subsection, a 3D boundary point exploration (3DBPE) method is proposed for dealing with the problem in which the corresponding point in the point cloud cannot be found for the edge pixels. B2B-DTL and its improved version are described at the end of this section. Next, in Section VI, we present the results of the experimental evaluation of the proposed method. Finally, in Section VII, we provide some concluding remarks.

This paper uses the following notations. The set of real numbers is represented by \mathbb{R} . The set containing real numbers and empty is written \mathbb{R}_\emptyset . The binary number set is given by $\mathbb{Z}_2 = \{0, 1\}$. The set of natural numbers is represented by \mathbb{N} . For a vector $a \in \mathbb{R}^n$, the i -th element a are denoted by a_i . A distance function d is defined by

$$d(a, b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \dots + (a_n - b_n)^2} \quad (1)$$

for arbitrary n dementional vectors a and b .

II. OVERVIEW OF PPF-MEAM

For comparison, the previous version of PPF-MEAM is briefly introduced in this section. The original PPF-MEAM can be divided into two parts: an offline database generation and an online pose estimation.

During the offline phase, we create a hash table for describing the target object. This table will be utilized as a reference for finding the object during the online phase. Normally, the data of the target object are given in the form of a CAD. After transforming the CAD model into a mesh file, the point cloud of the object can be obtained by generating random points on the mesh. The coordinate system of the model point cloud needs to be determined by referring to Σ_o , as shown in Fig. 6. Several visible point sets of the model from N viewpoints are then extracted. We call the N extracted visible point sets as N multiple appearance models. We set N to 6 based on experience. Next, we extract the boundary points from each appearance model based on the method provided in the point cloud library [50]. Each extracted boundary point set is called an edge appearance model, and B2B-TL descriptors are computed for each of these models. Because we utilize the values of the descriptors as the key for the hash table, point pairs that have the same value of the descriptor are inserted into the same bin.

During the online phase, a color camera and a 3D sensor are utilized to obtain an image and the point cloud and scene.

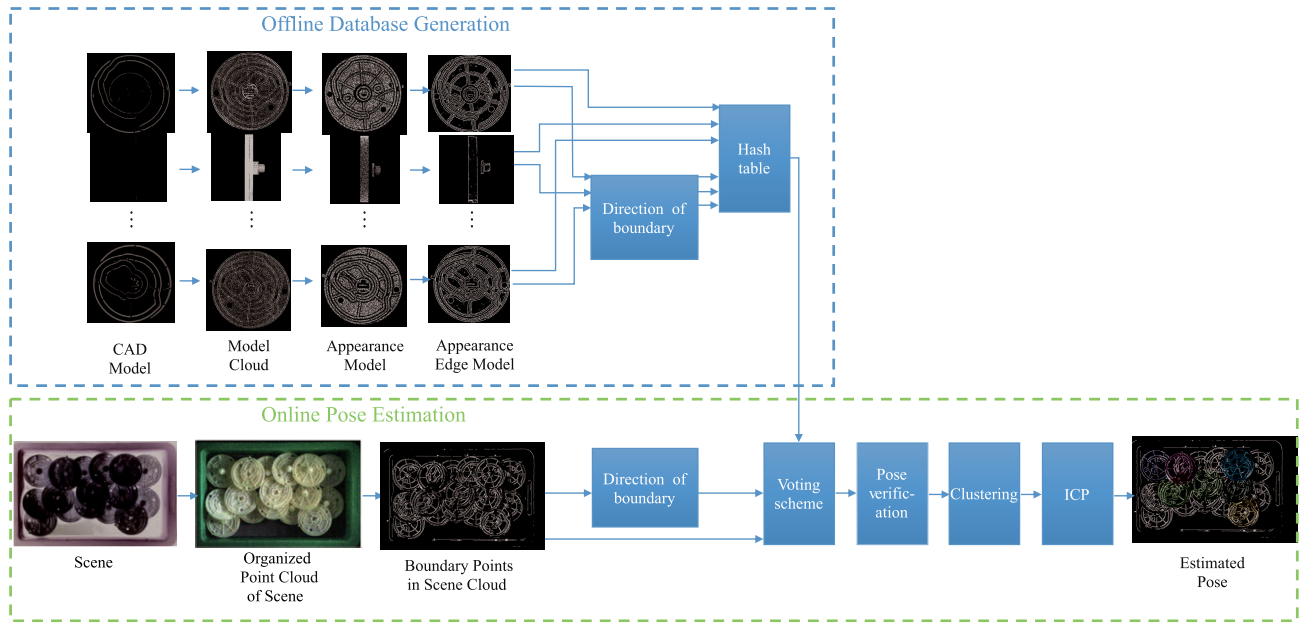


FIGURE 3. Pipeline of the original PPF-MEAM [33]. This method can be used to estimate the pose of industrial parts with high accuracy and short computation time. However, it may output the pose of an occluded object, which results in a failure in object picking.

Because the later processing necessitates the correspondence between each pixel in the image and each point in the point cloud, we calibrate the camera and 3D sensor to obtain their extrinsic parameters. The calibration program is provided by the SDK of the 3D sensor. After this calibration, we can obtain the mapping relation between each pixel in the captured image to the point in the point cloud. Because the extrinsic parameters of the camera and 3D sensor are known, we then extract 3D boundary points from the scene point cloud by determining their corresponding 2D edge pixels in the scene image. The 2D edge pixels are detected by applying the Canny edge detector [35] to the scene image. The boundary points of the occlusion-free objects are cropped from the entire scene of the point cloud. A Hough-like voting scheme is then conducted to obtain many coarse pose candidates, similar to the approach described by Drost *et al.* [21]. Some incorrect pose candidates are filtered out by conducting a pose verification, as proposed by Li and Hashimoto [51]. Pose verification using MEAM is applied to remove the incorrect pose candidates. For each model, a defined number of pose candidates can be obtained. The remaining pose candidates are sorted based on the pose verification score and refined using the ICP algorithm [52]. These refined pose candidates are sent to the next step in the bin picking system.

Although PPF-MEAM has shown a high-performance level in the pose estimation of industrial parts, it estimates poses of objects regardless of the occlusion of each object, which will likely result in a failure of the robotic picking of an occluded object in the bin.

III. PROPOSED METHOD

Occlusion-free PPF-MEAM is proposed to increase the robustness of the PPF-MEAM algorithm, which can estimate

the pose of only occlusion-free objects. As shown in Fig. 4, occlusion-free PPF-MEAM takes the input in the form of an RGB image and a 3D point cloud. The position and 2D feature points of multiple target parts that are not occluded are detected in a 2D image by forwarding the RGB image to a deep learning framework. The edge image is detected using the Canny edge algorithm [35]. Using the information of the object detection and edge image, the edge pixels of those parts that are not occluded are obtained. We then extract the boundary point cloud of the target parts using the edge pixels from the last step and the point cloud of the scene. Next, we obtain the 3D feature point using a pixel of the 2D feature point. Using the corresponding 3D feature point, we then input the boundary point cloud of each part individually into the PPF-MEAM, which is improved using the descriptor B2B-DTL.

A. OCCLUSION-FREE OBJECT DETECTION

Occlusion-free object detection includes two aspects, namely, finding occlusion-free objects and localizing them. In this study, object detection methods based on deep learning are considered for occlusion-free object detection for two reasons. First, deep learning-based methods [36] show higher performance than a traditional object detection method. This can be seen from the results of the ILSVRC competition. In particular, the recognition error rate of ResNet even exceeded that of the human eye by 5.1% [36]. Second, it is difficult to achieve occlusion detection using a traditional object detection method because it is challenging to define the occlusion for different types of objects. As a consequence, we need to develop a deep learning-based method to realize occlusion-free object detection.

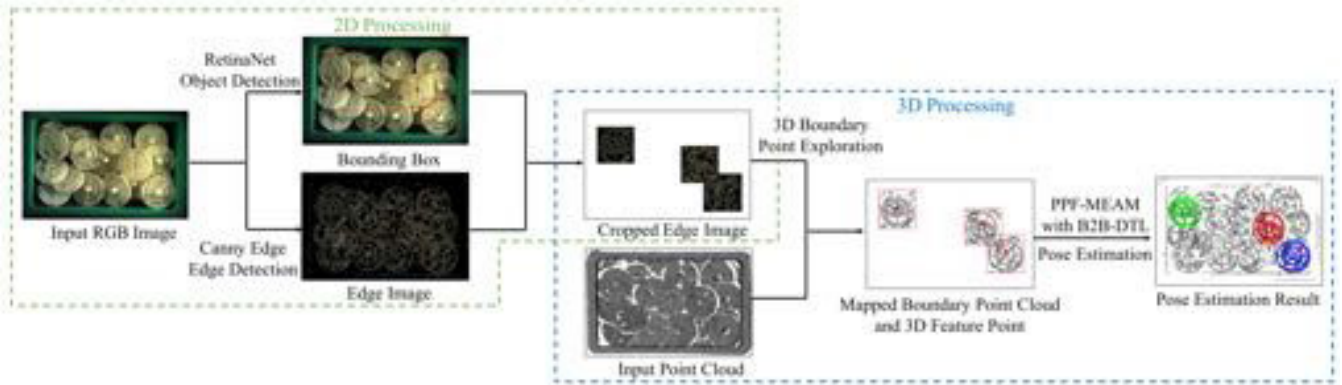


FIGURE 4. Pipeline of Occlusion-Free PPF-MEAM. The method estimates the pose of occlusion-free objects randomly piled up in a bin. Using a 2D process, the input RGB image of the scene is captured with a color camera. We then apply object and edge detection on this image using a modified RetinaNet algorithm and a Canny edge algorithm, respectively. Utilizing the results of the object and edge detection, the edges of the target parts are cropped. Please note that only those parts that can be easily grasped (occlusion-free) are detected. With a 3D process, the point cloud is taken using a 3D sensor. Using the input point cloud and cropped edge image, the boundary point cloud of the target parts is cropped. The 3D feature point is the corresponding point cloud of the pixel of a 2D feature point. The mapped boundary point cloud and 3D feature point are used as the input of the framework of the PPF-MEAM, which implements the B2B-DTL descriptor described in this study.

Deep learning-based object detection methods can be roughly divided into two categories, namely, one-stage and two-stage detectors. One-stage detectors [37]–[39] have an advantage of speed, although the accuracy is not as good as that of a two-stage detector [42]–[44]. The balance between speed and precision has always been a difficult question, and the emergence of RetinaNet [34] has improved this situation. RetinaNet is able to match the speed of the previous one-stage detectors while surpassing the accuracy of all existing state-of-the-art two-stage detectors. Lin *et al.* [34] compared RetinaNet with other previous networks. The results are shown in Table 1. We found that RetinaNet achieves an excellent balance between the inference time and AP. Therefore, with our approach, we use RetinaNet as the regression network for object detection.

TABLE 1. Comparison of Performance for Object Detection Method.

Method Name	AP	Time [ms]
YOLOv2 [39]	21.6	25
SSD321 [38]	28.0	61
DSSD321 [40]	28.0	85
R-FCN [41]	29.9	85
SSD513 [38]	31.2	125
DSSD513 [40]	33.2	156
FPN FRCN [46]	36.2	172
RetinaNet-50-500 [34]	32.5	73
RetinaNet-101-500 [34]	34.4	90
RetinaNet-101-800 [34]	37.8	198

1) MODIFICATION OF RetinaNet

In this section, we detail the usage and modification of RetinaNet [34]. The architecture of RetinaNet used in the proposed method is shown in Fig. 5.

RetinaNet is an end-to-end detection network framework. Compared with fast R-CNN, SSD, and other detection

networks [38], [43], [44], RetinaNet has the advantages of fast detection and high precision. It consists of a backbone network and two subnets. The backbone network is responsible for extracting the feature map from the input image. The extracted feature maps are fed into two subnets. The first subnet is responsible for classification. The second subnet performs a regression of the bounding box. A feature extractor, ResNet-50 or ResNet-101, is used to initially extract a feature map from the input image. The feature pyramid network builds five feature maps based on the top of the extractor. This structure achieves excellent performance in detecting large and small-sized objects at the same time. The classification subnet predicts the probability of the existence of objects at each spatial location. In our case, we only predict whether the object is occluded, that is, there are two categories. Box regression subnet has a similar structure as the classification network and the subnet is responsible for outputting the offset of the bounding box relative to each anchor box. In this study, a 2D point subnet used to regress the position of a 2D feature point, which is applied for a 3D boundary point exploration and calculating the B2B-DTL, has been attached to the original architecture.

2) 2D POINT SUBNET

A 2D point subnet is a small FCN attached to each FPN level, similar to the classification and box regression subnets. The features extracted by an FPN are fed to the point subnet for regressing the offset from each anchor box to a feature point of the object. The subnet contains four 3×3 Conv layers, where each layer is activated using ReLU. After the four conv layers, the output layer is a 3×3 Conv layer with 2A filters. A Sigmoid is used as the activated function.

3) POINT ENCODING

Inspired by the design of a box regression, we do not directly regress the coordinates of the point but rather the offset.

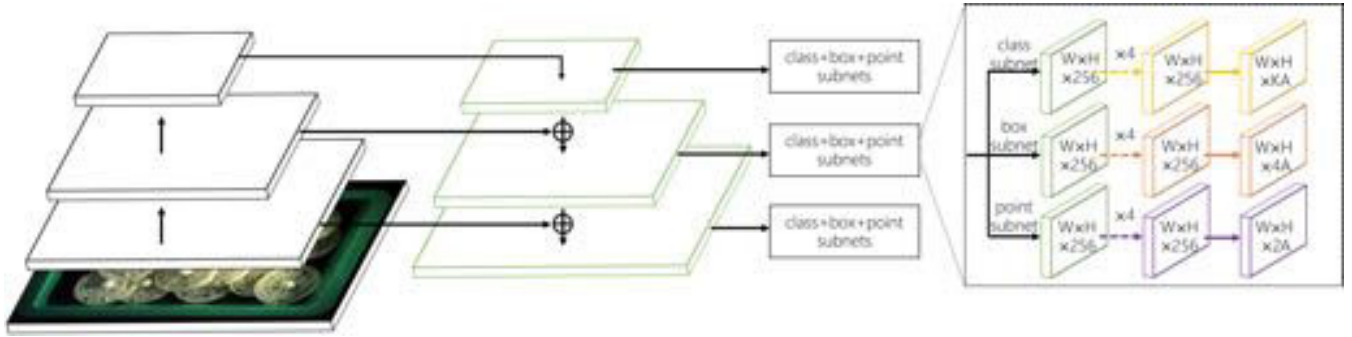


FIGURE 5. Network architecture of RetinaNet [34]. RetinaNet consists of a backbone network and three task-specific subnetworks. The backbone is utilized for computing a feature map over an entire input image. A feature pyramid network is built on the top of ResNet-50. The first subnet conducts classification on the backbone output. The result of the classification is the probability of the presence of K object classes at each spatial position for each anchor A . The second subnet applies a convolution bounding box regression. It outputs the object localization with respect to the anchor boxes if an object exists. The third subnet conducts a convolution 2D feature point regression. It outputs the position of the 2D feature point in the image.

The offset of the feature points (two coordinates) was designed as follows:

$$\begin{aligned} c_x &= (x_c - x_a)/w_a, c_y = (y_c - y_a)/h_a, \\ c_x^* &= (x_c^* - x_a)/w_a, c_y^* = (y_c^* - y_a)/h_a, \end{aligned} \quad (2)$$

where x_a, y_a, w_a, h_a denote the center coordinates and the width and height of the anchor box. In addition, x_c, y_c are the center coordinates, and x_c^*, y_c^* are the predicted center coordinates. Each anchor corresponds to a point, and thus the output of a point subnet is a $2A$ linear result.

4) LOSS FUNCTION OF 2D POINT SUBNET

A smooth L1 loss is used as a loss function in the point subnet. For each anchor, the point subnet predicts two values, c_x^* and c_y^* . Thus, the predicted point is represented as $C^* = (c_x^*, c_y^*)$, and the ground truth is denoted as $C = (c_x, c_y)$. The regression loss L_P is defined as follows:

$$L_P = \sum_{i \in \{x,y\}} \text{smooth}_{L1}(C_i^* - C_i), \quad (3)$$

where,

$$\text{smooth}_{L1}(x) = \begin{cases} 0.5|x|^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \quad (4)$$

5) TRAINING DATA

In this study, we used four objects, which are the same industrial parts described in the previous section, i.e., parts A, B, C, and D. We took approximately 250 images of each object. In each scene, one type of part is piled up randomly in a bin. The position of the objects in each image is described by bounding boxes and labeled manually. In addition, the 2D feature point is also labeled manually. Examples are shown in Fig. 7 and 8.

A 2D feature point is defined using the following steps. Here, we take the example of Part A.

- 1) Define the coordinate system of the model point cloud. Please note that we are using the model point cloud as

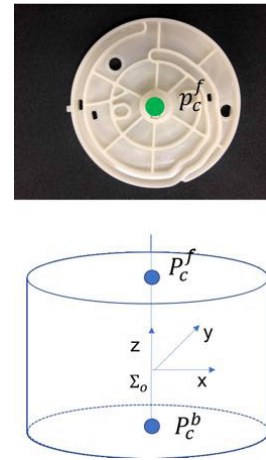


FIGURE 6. Definition of 2D feature point. We first define an object coordinate system Σ_o . 3D feature points, P_c^f and P_c^b , are the points of intersection of the z -axis and surface of the object. A 2D feature point, P_c^f , is defined as P_c^f in the 2D images.

- an input in this step. The origin of this coordinate system is the centroid of the model point cloud. Determining the direction of the x -, y -, and z -axes requires a PCA analysis of the model point cloud. Three eigenvalues obtained through a PCA analysis are sorted from largest to smallest. Three corresponding eigenvectors of these eigenvalues are used as directions of the x -, y -, and z -axes. This coordinate system Σ_o is shown in Fig. 6.
- 2) Define the 3D feature point. Here, we use a cylinder to represent the point cloud of Part A, as shown in Fig. 6. When we extend the z -axis of Σ_o , the cylinder's upper and lower surfaces will then intersect the z -axis, resulting in points P_c^f and P_c^b . Because the actual point cloud is sparse, we will use the point in the point cloud closest to the two points P_c^f and P_c^b as the 3D feature point.
- 3) Define a 2D feature point. On the grounds that we have obtained 3D feature points, 2D feature points can then be obtained by mapping to the 2D space. Specifically, we project the point cloud of the upper and lower surfaces onto the xy -plane, and the pixel corresponding to the 3D feature point as a 2D feature point.



FIGURE 7. Training data. These data are made by labeling the bounding box and 2D feature point (green dots) of objects that are not occluded.

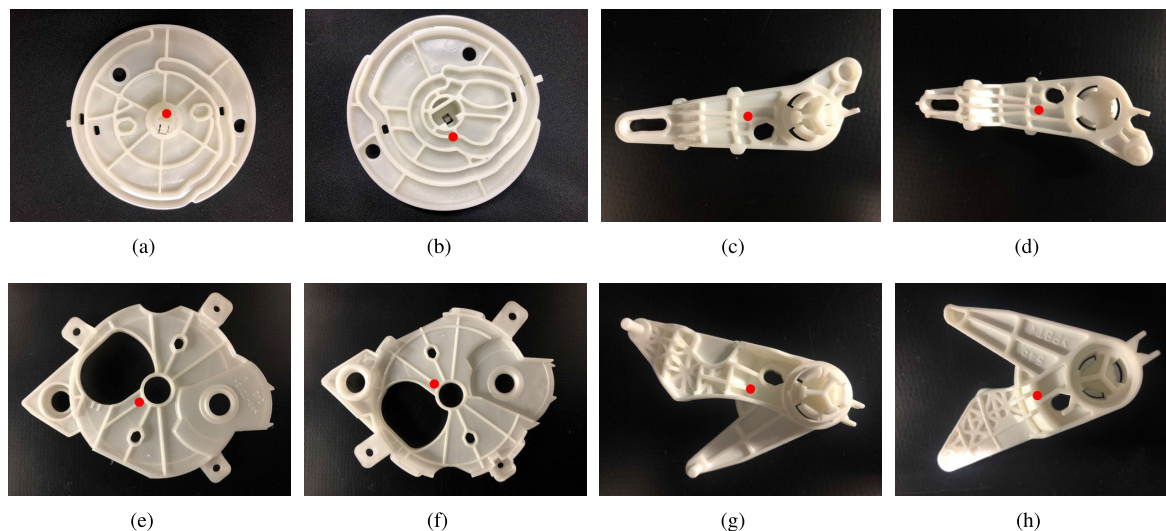


FIGURE 8. 2D feature point of each type of part. Because flat objects such as parts A, B, C, and D tend to show either a front side or back side in a randomly piled up stack, we only define the feature point for these two sides.

A definition of the 2D feature point of each part is shown in Fig. 8.

The position of the 2D feature points can be defined manually as well because the variation in the shapes of the different objects is extremely large. The aforementioned definition can be referenced as a way to find this position. Here, 2D feature points are used directly in the 3D boundary point exploration. In addition, mapping to a 3D point will be applied to obtain a 3D feature point in the scene, which is used for calculating the B2B-DTL. Because flat objects such as Parts A, B, C, and D, tend to show either a front or backside in a randomly piled up stack, we only define the feature points for these two sides. In this research, we focused on flat objects; cube-like and curvy objects are not considered.

As a unique proposition, we only label the objects that are not occluded (occlusion-free) in the scene. As mentioned earlier, only the pose of the objects that are not occluded is desired for the robotic bin-picking system. There are two reasons for this. First, there is likely no space for the gripper to grasp the occluded object. Second, even if the robot can pick up an object under other objects, during the picking motion, the grasped object will collide with other objects. Such collision results in changes in the pose of the object with respect to the gripper, which probably results in a failure

of the following placement task, particularly for heavy target objects.

We use 200 images as the training set and 50 images as the test set. To conquer the problem of a small number of training images, data augmentation, i.e., horizontal/vertical flipping and shifting, has been utilized. We trained each part of our model for 30 epochs, and each epoch has 1000 steps with a batch size of 1. We implemented our code under the Keras framework using TensorFlow as the back end. This takes approximately 5 h using an Nvidia GeForce GTX 1080 GPU. The Resnet-50 model pre-trained on COCO data [47] is implemented to decrease the training time to reach the convergence point.

6) VALIDATION EXPERIMENT

To evaluate the performance of RetinaNet, we conducted several validation experiments. Some representative inference results are shown in Fig. 9. The accuracy of the object detection is shown in Table 2. The confidence threshold is used here. For each type of object, the accuracies of the top-1 through top-3 results are calculated. If the confidence of the result is lower than the threshold, the result will not be output. Please note that, for the test set, we only labeled the objects without an occlusion. Even if the object detection

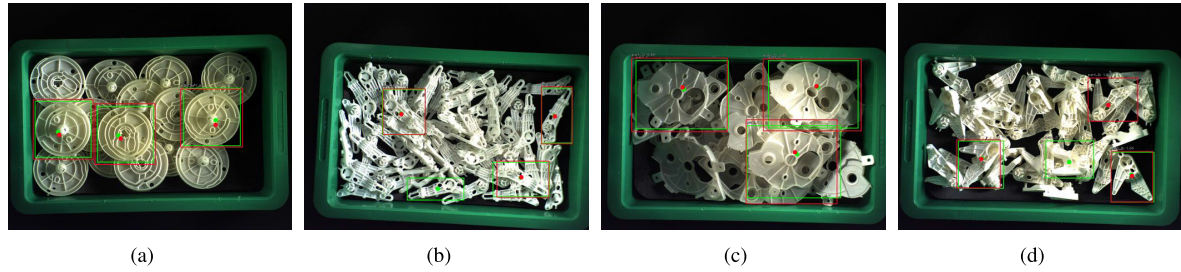


FIGURE 9. Result of occlusion-free detection. Inference results are shown in red; the ground truth is shown in green. Please note that only objects without an occlusion are detected.

TABLE 2. Accuracy of Occlusion-Free Object Detection.

Part	Threshold	top 3	top 2	top 1
Part A	0.5	98.46%	100.00%	100.00%
	0.8	97.66%	100.00%	100.00%
Part B	0.5	85.71%	88.24%	89.86%
	0.8	85.86%	88.15%	89.86%
Part C	0.5	79.82%	82.42%	92.00%
	0.8	82.35%	85.06%	92.00%
Part D	0.5	84.68%	87.80%	88.10%
	0.8	84.62%	88.31%	88.10%

is correct when an occlusion is present, it would not be considered a correct detection.

In addition, we computed the average 2D feature point error, as shown in Table 3.

TABLE 3. Error of Feature Point [Pixels].

Parts	Threshold	top 3	top 2	top 1
Part A	0.5	7.87	8.04	8.54
	0.8	7.87	8.04	8.54
Part B	0.5	7.44	7.15	7.57
	0.8	7.44	7.15	7.57
Part C	0.5	10.87	10.61	11.12
	0.8	10.05	10.73	11.12
Part D	0.5	10.00	10.03	9.34
	0.8	10.16	10.23	9.34

The AP rate in each scene is shown in Table 4.

TABLE 4. AP Rate.

threshold	Part A	Part B	Part C	Part D
0.5	96.14%	68.33%	88.19%	69.21%

According to the result of the validation experiment, we consider the performance of this network to be sufficiently high to be used in occlusion-free PPF-MEAM.

B. 3D BOUNDARY POINT EXPLORATION (3DEPE)

Owing to the calibration error between the camera and 3D sensor, the corresponding point in the point cloud

occasionally cannot be found for the edge pixel, as shown in Fig. 10. A lack of boundary points results in an incorrect pose estimation result.

To solve this problem, a 3D boundary point exploration is proposed. If the corresponding 3D edge points cannot be found, neighbor pixels of the edge pixels will be considered as candidates for finding 3D edge points. Neighboring pixels along the line connecting from the edge pixel to the feature point are explored, as shown in Fig. 11. The pseudocode is shown in Algorithm 1. Here, $I_{edge} \in \mathbb{Z}_2^{m \times n}$ is the binary edge image of the scene, which is cropped for one part. $I_{xyz} \in \mathbb{R}^{m \times n \times 3}$ is a 3-channel image consisting of the point cloud of the same part of I_{edge} . The $(i, j, 1)$, $(i, j, 2)$, $(i, j, 3)$ -th elements of I_{xyz} represent x , y , and z coordinates of the 3D point corresponding to the (i, j) -th element in I_{edge} . If the calibration error between the camera and 3D sensor is large, the corresponding pixel in I_{xyz} will be empty. L_{step} is a parameter that determines the length of the exploration.

C. BOUNDARY-TO-BOUNDARY-USING-DIRECTIONAL-TANGENT-LINE (B2B-DTL)

In this study, a descriptor called B2B-DTL [48], [49] was improved and utilized in the PPF-MEAM pipeline instead of B2B-TL. B2B-DTL [48] is an improved PPF based on B2B-TL [33]. The definition of B2B-DTL is shown in Fig. 12, and the descriptor was designed to find the corresponding point pairs between the scene and model. Using these corresponding point pairs, a 6D pose can be computed. As mentioned by Liu *et al.* [33], for some industrial parts, points along the boundary of the objects have more essential information than those on the surface, as shown in Fig. 1. In addition, the number of boundary points of an object is much less than the number of surface points. Therefore, the boundary points of an object are preferred for a pose estimation while pursuing a short computation time. When we use points on the boundary, we use the direction of the tangent line that passes through these points as the orientation instead of the normal direction because one boundary point has an infinite number in this direction.

Computing the tangent line among the boundary points is easy when using the method proposed for the original PPF-MEAM pipeline [33] although choosing which direction to use along the tangent line is difficult because each tangent line can represent two opposite directions. For the

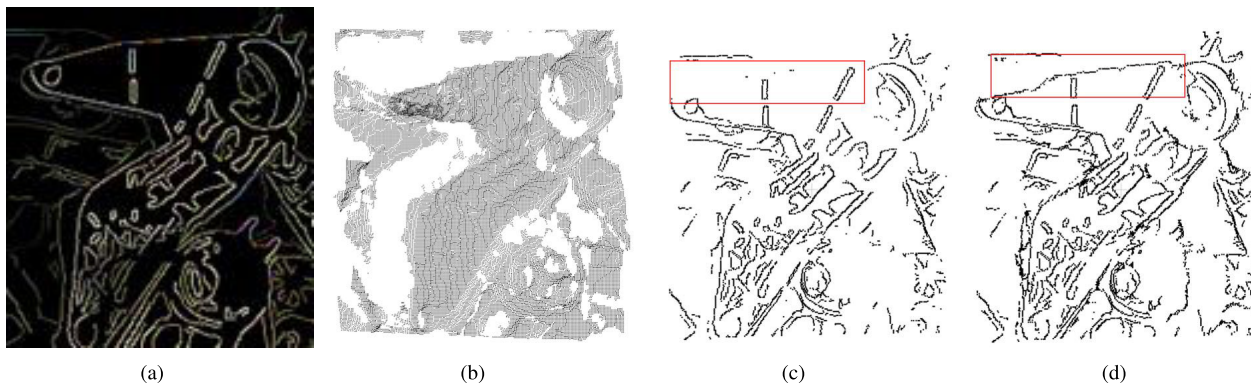


FIGURE 10. Result of 3DBPE. (a): edge image, (b): captured point cloud. (c): captured point cloud. (d): point cloud refined by 3DBPE. Using (a) and (b), (c) is obtained by mapping edge pixels to a point cloud. As an error of the calibration, the boundary point cloud in the red box is not completely obtained. (d): Boundary point cloud obtained by mapping edge pixels to point cloud with 3DBPE. Compared with (c), the boundary point cloud in a red box has been sufficiently obtained.

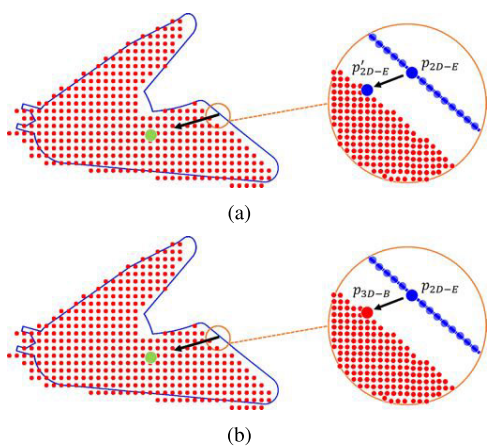


FIGURE 11. Finding corresponding points in 3DBPE. (a): the deviation between the detected edge (blue) and point cloud (red) owing to the calibration error between the camera and 3D sensor. If the corresponding 3D boundary point cannot be found for a point P_{2D-E} along the 2D edge, a neighbor point P'_{2D-E} in the image will be searched along the line (black) from the point along the 2D edge to the 2D feature point (green). (b): the 3D point P_{3D-B} corresponding to P_{2D-E} is extracted as the corresponding 3D point of P_{2D-E} . This process will be conducted for those points along the 2D edge that do not have corresponding 3D points.

corresponding point in the model and scene, it is difficult to guarantee that the same direction will be chosen. For example, the point P_i in Fig. 13 (a) can use the direction of \bar{n}_i , whereas the P_i in Fig. 13 (b) uses the direction of $-\bar{n}_i$. This leads to an incorrect matching and results in an incorrect pose candidate. This problem can be solved using the clockwise direction, as mentioned in [48], which guarantees that the direction is chosen, as shown in Fig. 13. The point P_c in the aforementioned study is obtained based on the center of the bounding box, which brings about a certain disadvantage, i.e., with a change in the size of the detected bounding box, the position of P_c also changes. During the off-line phase, P_c is determined using an ideal bounding box. Therefore, we can imagine that if the detected bounding box is slightly smaller than the ideal bounding box, P_c in Fig. 13 (a) and 13 (b) is at

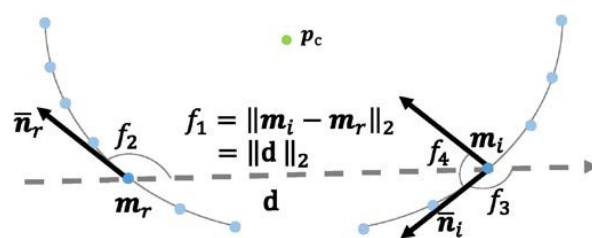


FIGURE 12. Definition of B2B-DTL. The m_r is the reference point and m_i is the referred point. The \bar{n}_r and \bar{n}_i represents the clockwise direction of the tangent line that passes through these two points. Here, P_c is a 3D feature point. Because the range of the angular component is from $[0, \frac{\pi}{2}]$ to $[0, \pi]$ (where f_2 and f_4 are obtuse angles), the success rate and speed of the computation of the pose estimation can be increased.

a different position with respect to the part. A small bounding box will likely be detected if the tilted part is occluded.

As an important improvement in this study, the 3D feature point P_c , i.e., the corresponding 3D point of the 2D feature point in the image, is used to find the clockwise direction of the tangent line. If there is no corresponding 3D point in the scene for the 2D feature point, the nearest pixel that has a corresponding 3D point is searched, and the 3D point is used as the 3D feature point. During the process of creating a hash table, we can use the defined 3D feature point for calculating the features.

After we obtain the 3D feature point, we compute the angles from $P_i P_{feature}$ to the \bar{n}_i and $-\bar{n}_i$. If this angle is larger than a threshold, it is labeled as the counterclockwise angle; otherwise, it is the clockwise angle. The direction vectors that yield the clockwise angle are used as the orientation of points for computing the B2B-DTL. The B2B-DTL is defined by the following:

$$F_{B2B-DTL} = (f_1, f_2, f_3, f_4)^T = (\|d\|_2, \angle(\bar{n}_r, d), \angle(\bar{n}_i, d), \angle(\bar{n}_i, \bar{n}_r))^T, \quad (5)$$

where we use the same notation as that applied for describing B2B-TL. Unlike using the sharp angle for the angular elements in B2B-TL, the range of the angular component

Algorithm 1 3D Boundary Point Exploration**Data:**

$I_{\text{edge}} \in \mathbb{Z}_2^{m \times n}$: A binary edge image of the scene which is mainly cropped for one part. If the edge exists, the corresponding pixel equals one.

$I_{\text{xyz}} \in \mathbb{R}_{\emptyset}^{m \times n \times 3}$: A three-channel image, where the $(i, j, 1)$, $(i, j, 2)$, and $(i, j, 3)$ -th elements of I_{xyz} represent x , y and z coordinates of the 3D point in (i, j) -th pixel in the corresponding image, respectively. If the edge does not exist, the corresponding elements equal to empty.

$P_{\text{feature}} \in \mathbb{N}^2$: A coordinate of a Feature Point in I_{edge} .

Result: 3D points set \mathbb{S}_{pts} which consists of the boundary point cloud of one part

$\mathbb{S}_{\text{pts}} \leftarrow \emptyset$;

$p_{\text{tmp}} = [0, 0, 0]^T$;

for $i \leftarrow 1$ **to** m **do**

for $j \leftarrow 1$ **to** n **do**

if $I_{\text{edge}}(i, j) \neq 0 \cap I_{\text{xyz}}(i, j, 1) \neq \emptyset$ **then**

$p_{\text{tmp},1} \leftarrow I_{\text{xyz}}(i, j, 1)$;

$p_{\text{tmp},2} \leftarrow I_{\text{xyz}}(i, j, 2)$;

$p_{\text{tmp},3} \leftarrow I_{\text{xyz}}(i, j, 3)$;

$\mathbb{S}_{\text{pts}} \leftarrow \mathbb{S}_{\text{pts}} \cup \{p_{\text{tmp}}\}$;

end

if $I_{\text{edge}}(i, j) \neq 0 \cap I_{\text{xyz}}(i, j, 1) = \emptyset$ **then**

$\Delta y \leftarrow \text{Round} \left(\frac{P_{\text{feature},1-i}}{d(P_{\text{feature},[i,j]^T})} \right)$;

$\Delta x \leftarrow \text{Round} \left(\frac{P_{\text{feature},2-j}}{d(P_{\text{feature},[i,j]^T})} \right)$;

for $k \leftarrow 1$ **to** L_{step} **do**

$i' \leftarrow i + k\Delta y$;

$j' \leftarrow j + k\Delta x$;

if $I_{\text{xyz}}(i', j', 1) \neq \emptyset$ **then**

$p_{\text{tmp},1} \leftarrow I_{\text{xyz}}(i', j', 1)$;

$p_{\text{tmp},2} \leftarrow I_{\text{xyz}}(i', j', 2)$;

$p_{\text{tmp},3} \leftarrow I_{\text{xyz}}(i', j', 3)$;

$\mathbb{S}_{\text{pts}} \leftarrow \mathbb{S}_{\text{pts}} \cup \{p_{\text{tmp}}\}$;

end

end

end

end

end

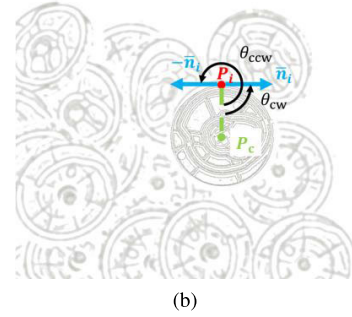
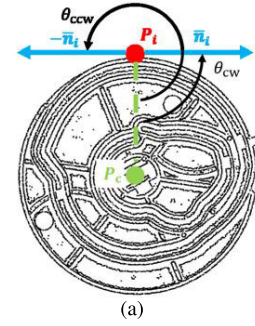


FIGURE 13. Clockwise direction among the two directions along the tangent line is utilized. Point P_i in the figure has two directions \bar{n}_i and $-\bar{n}_i$ along its tangent line. Its 3D feature point P_c is also known. Two angles θ_{ccw} and θ_{cw} from $P_i P_c$ to \bar{n}_i and $-\bar{n}_i$ can be obtained. These two angles are utilized to determine which direction is clockwise. Because θ_{cw} is less than 180° , \bar{n}_i is used as the orientation of P_i for computing the B2B-DTL. The clockwise direction along the tangent line is found for both the model (a) and the scene (b).

IV. EVALUATION EXPERIMENT

A. REAL SCENE DATA

We evaluated an occlusion-free PPF-MEAM using data of real scenes, the results of which are described herein. To validate this method in a practical application, a robotic picking experiment is also conducted. The original PPF [21] and the original PPF-MEAM with the B2B-TL are compared to show the advantages of the proposed method.

Python was implemented for object detection, and C++ was used for pose estimation. The programs were run on a single computer with an Intel Core i7 6950X CPU, 32 GB of RAM, and a GTX1080 GPU. To determine the best performance of the proposed method for a real application, the pose estimation programs used in this experiment were accelerated by applying the OpenMP frame. We used an Ensenso N35 3D sensor and an iDS USB 3 uEye 2D camera. Two LEDs were set near the bin to provide a stable lighting environment.

We tested 20 scenes for each of the four types of parts. For each scene, Parts A, B, C, and D were randomly piled up inside a bin. During the 2D process, τ_t was set to 0.5. The maximum number of outputs was set to 4. This means that the cropped point cloud of four parts at most was input to the PPF-MEAM framework. The maximum number was defined through experience. In a 3D boundary point exploration, L_{step} was set to 3 for part D and zero for the other parts. During the 3D process, we used the same notations as in the original PPF-MEAM [33]. Here, τ_d resulting in $d_{\text{dist}} = \tau_d \times D$ was

changes from $[0, \frac{\pi}{2}]$ to $[0, \pi]$. As mentioned in [48], this leads to a shorter computation time and fewer incorrect pose candidates.

Because flat objects such as Parts A, B, C, and D, tend to show either the front or backside in a randomly piled-up stack, we only define the 2D and 3D feature points for these two sides for each type of object. During the offline phase, we compute the B2B-DTL for the point pairs in each MEAM, as mentioned previously. For some MEAMs, there are no corresponding feature points defined. In this case, the 3D centroid of the point cloud will be used as the feature point for the MEAM to compute the B2B-DTL.

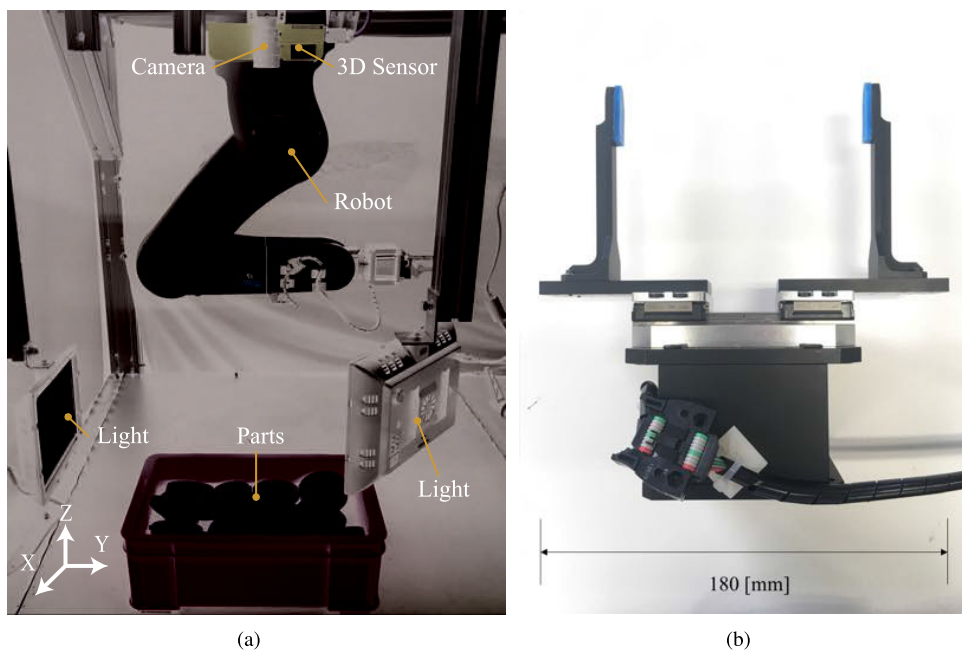


FIGURE 14. Robotic bin picking system utilized to validate the proposed method. A color camera and 3D sensor were fixed on the frame above the parts. To provide a stable lighting condition, two light-emitting diodes (LEDs) were installed on both sides of the box. The gripper used for the picking task is shown in (b).

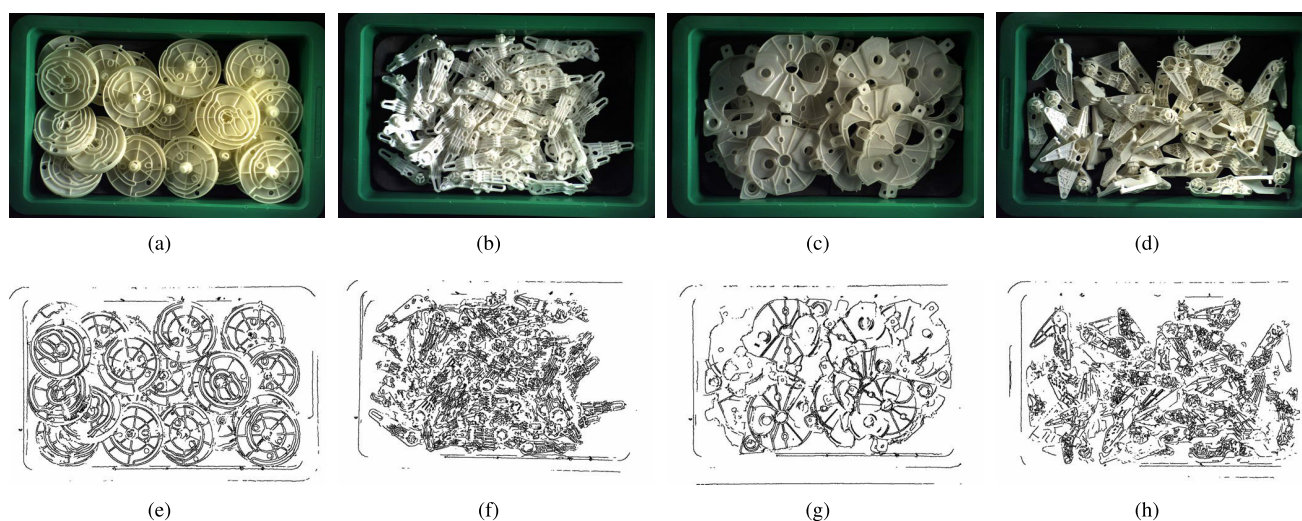


FIGURE 15. (a)–(d): images of real example parts. (e)–(h): scene cloud containing only boundary points.

equal to 0.07, where D is the maximum distance between two points on the object. We set d_{angle} to 3.6° based on our experience. To balance between the recognition rate and the computation time, P_{ref} and $P_{referred}$ determined the percentage of scene boundary points used as reference and referred points, respectively. We set $P_{referred}$ to 20% and P_{ref} to 10%, 10%, 10%, and 20% for Parts A, B, C, and D, respectively. For the other parameters, we used the same value as shown in the original study on PPF-MEAM.

1) QUANTITATIVE EVALUATION

We output three pose results in each scene for Parts A, B, and D and two pose results for Part C. The reason for setting the

output number in such a way is based on the average number of parts without occlusion in each scene. The correctness of the resulting pose is manually determined. We counted the number of true positives, and the recognition rate is the number of true positives over the number of output poses.

The speed and recognition rate of the proposed method for each type are presented in Tables 5 and 6, respectively. The occlusion-free rate is presented in Table 7. This is the number of true positives of parts that are not occluded over the number of all true positives. Some examples are shown in Figs. 15 and 16. We rendered the top pose results using red, green, and blue in order. We compared our method with the original PPF [21] and original PPF-MEAM.

TABLE 5. Speed of the algorithms for real scenes (ms/scene).

Methods	Part A	Part B	Part C	Part D	Average
Original PPF [21]	2443	3452	1757	1527	2294
Original PPF-MEAM	1454	1370	1353	1140	1329
Proposed Method	1076	982	974	988	1005

TABLE 6. Recognition rate of the algorithms for real scenes.

Methods	Part A	Part B	Part C	Part D	Average
Original PPF [21]	15.0%	85.0%	75.0%	95.0%	67.5%
Original PPF-MEAM	95.0%	96.7%	97.5%	96.7%	96.5%
Proposed Method	96.6%	96.6%	97.4%	98.2%	97.2%

TABLE 7. Occlusion-Free rate of the algorithms for real scenes.

Methods	Part A	Part B	Part C	Part D	Average
Original PPF [21]	55.6%	43.1%	56.7%	36.2%	47.9%
Original PPF-MEAM	86.0%	39.7%	50.0%	50.0%	56.4%
Proposed Method	100%	84.2%	86.5%	71.9%	85.7%

For parts similar to Part A, occlusion-free PPF-MEAM shows an estimation with a high recognition rate, whereas the original PPF does not. This is due to the insufficient data captured by the 3D measurement sensor, as shown in Fig. 1. The original PPF can estimate the position of Part A. However, the original PPF cannot estimate the rotation around z axis in the coordinate system of Part A because the captured point cloud does not have sufficient information near the ridges in Part A.

Compared with the original PPF-MEAM, the proposed method is faster while maintaining almost the same recognition rate. The small difference in the success rate between the original PPF-MEAM and the proposed method for Parts C and D is caused by the different numbers of total output. For the proposed method, if RetinaNet detects only one object in the scene without an occlusion, there is one final pose estimation result. However, the number of outputs of the original PPF-MEAM is the defined number regardless of whether the parts are occluded. Considering the occlusion-free rate, the proposed method outperforms the other two methods. A high occlusion-free rate is considered to improve the success rate of a grasping task over the use of the other two methods.

2) CONTRIBUTION OF 3D BOUNDARY POINT EXPLORATION

To show the effectiveness of the 3D boundary point exploration, an experiment taking Part D as the example is conducted. During this experiment, the same image and a point cloud of the scene is used for the proposed method with and without 3DBPE. As shown in Table 8, we can increase the success rate by approximately 8% while maintaining a similar computation time.

B. BIN-PICKING SYSTEM PERFORMANCE

To validate the ability of the occlusion-free PPF-MEAM based on a practical application, a robotic picking experiment

TABLE 8. Contribution of 3D Boundary Point Exploration.

	Computation Time [ms]	Recognition Rate
Without 3DBPE	979	91.4%
With 3DBPE	983	98.2%

TABLE 9. Pickup success rate for Part A.

	Total number of trials	Success	Failure	Success Rate
OF-PPF-MEAM	64	60	4	93.8%
PPF-MEAM	63	49	14	77.8%

was conducted. The manipulator used in this experiment was a VS-068 from DENSO. An Ensenso N35 and an iDS USB 3 uEye were utilized as the 3D sensor and 2D camera, respectively. The gripper applied was an ESG 2 Series from Taiyo Co. Two LEDs were set near the bin to provide a stable lighting environment.

As shown in Fig. 15, multiple versions of Part A were piled up randomly in the bin. In addition, P_{ref} and $P_{referred}$ were set to 10% and 20%, respectively. Six edge appearance models were extracted and used to construct the hash table. RetinaNet output four object positions in the image at most. After applying PPF-MEAM, there were three output poses at most. The top-three pose candidates were output for each scene. Twelve parts were randomly piled up in the bin. The robot picked up these parts until no parts remained. Once the picking task failed, the pose estimation and picking task were conducted again. If there were no results for the grasp planning section, the objects were randomly shuffled. We tested five bins with parts, as shown in Table 9, and achieved a success rate of 93.8% for 64 trials, with an average computation time of approximately 860 ms for pose estimation. During this experiment, if the grasped part was occluded by other parts, we considered the trial as having failed.

Although 60 parts were successfully picked up during 63 trials using PPF-MEAM, and during the picking motion, the grasped object collided with another object 11 times. During this experiment, we considered this as a failed trial, which differs from the approach used in [33]. This means that we set a higher requirement for a successful picking task. For example, if the object is heavy, a collision will change the pose of the object with respect to the robot hand, resulting in a failure of a high-precision placement task. When we applied the occlusion-free PPF-MEAM during a picking task, it achieved a success rate of 100% in grasping objects without occlusions. However, because the entire system has no function in avoiding a collision, the opening gripper touched the other parts before grasping the target part four times. Because those parts were under the target, touching those parts changed the position of the target part, which led to the failed picking task. In the future, if collision avoidance can be integrated into this picking system, the success rate of the occlusion-free PPF-MEAM can be improved.

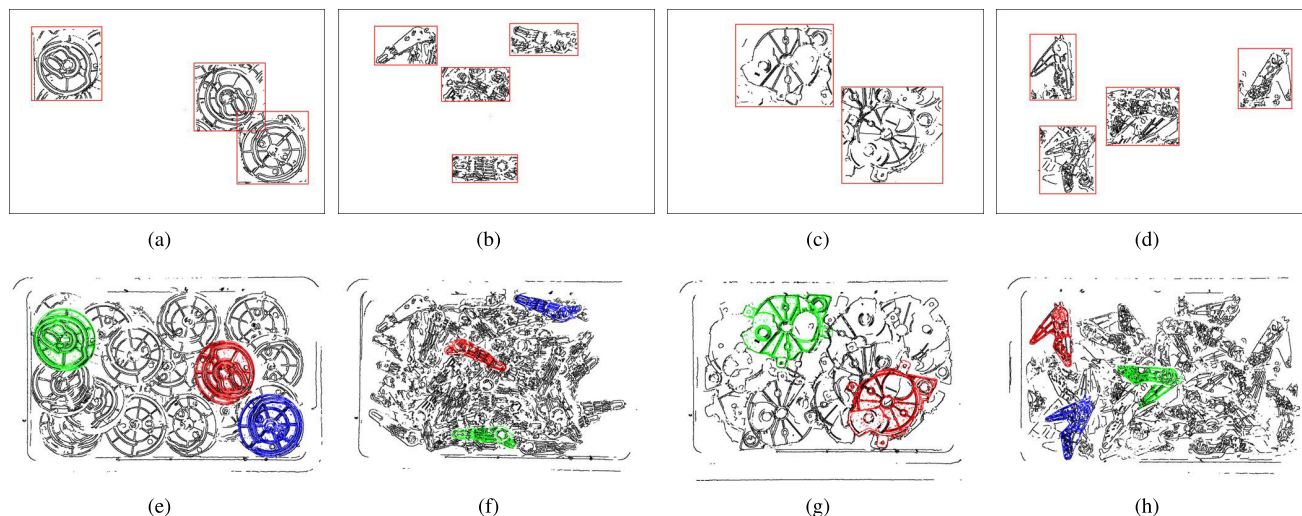


FIGURE 16. (a)–(d): cropped point cloud of objects without occlusion. (e)–(h): pose estimation results. The point cloud of the model is transformed into the scene space utilizing the pose results and rendered with different colors. These colors indicate the recommended picking order of the parts after considering the occlusion. Models are rendered as red, green, and blue.

V. CONCLUSION

In this study, a new pose estimation pipeline for robotic bin-picking called occlusion-free PPF-MEAM was proposed, in which a 3D pose estimation of occlusion-free objects is achieved by applying PPF-MEAM to a 2D image.

Occlusion-free PPF-MEAM uses deep learning-based 2D object detection and a non-machine learning-based 3D pose estimation method to estimate the pose of randomly piled industrial parts. The deep learning framework implemented on the proposed pipeline was modified for detecting occlusion-free objects and outputting 2D feature points. An algorithm called a 3D boundary point exploration (3DBPE) was also proposed for extracting sufficient boundary points in the point cloud using the edge pixels of the image in the scene. In addition, the descriptor, a B2B-DTL PPF, was improved to overcome some of the shortcomings of the previous version.

The main advantage of our method is that it increases the success rate of bin-picking owing to the recognition of occluded-free objects. More importantly, the 3DBPE algorithm and the improved B2B-DTL descriptor make the method more robust to a defective boundary point cloud and shorten the computation time. In the future, we will focus on those cases in which multiple types of parts are present for handling of more complex scenes.

ACKNOWLEDGMENT

(Diyi Liu and Shogo Arai are co-first authors.)

REFERENCES

- [1] Y. Gao, T. Cheng, Y. Su, X. Xu, Y. Zhang, and Q. Zhang, "High-efficiency and high-accuracy digital image correlation for three-dimensional measurement," *Opt. Lasers Eng.*, vol. 65, pp. 73–80, Feb. 2015.
- [2] N. Chiba, S. Arai, and K. Hashimoto, "Feedback projection for 3D measurements under complex lighting conditions," in *Proc. Amer. Control Conf. (ACC)*, May 2017, pp. 4649–4656.
- [3] B. Tippetts, D. J. Lee, K. Lillywhite, and J. Archibald, "Review of stereo vision algorithms and their suitability for resource-limited systems," *J. Real-Time Image Process.*, vol. 11, no. 1, pp. 5–25, Jan. 2016.
- [4] S. Arai, Y. Iwatani, and K. Hashimoto, "Fast sensor scheduling with communication costs for sensor networks," in *Proc. Amer. Control Conf.*, Jun. 2010, pp. 295–300.
- [5] S. Arai, Y. Iwatani, and K. Hashimoto, "Fast sensor scheduling for spatially distributed sensors," *IEEE Trans. Autom. Control*, vol. 56, no. 8, pp. 1900–1905, Aug. 2011.
- [6] S. Zhang, D. Van Der Weide, and J. Oliver, "Superfast phase-shifting method for 3-D shape measurement," *Opt. Exp.*, vol. 18, no. 9, pp. 9684–9689, 2010.
- [7] S. Zhang, "Recent progresses on real-time 3D shape measurement using digital fringe projection techniques," *Opt. Lasers Eng.*, vol. 48, no. 2, pp. 149–158, Feb. 2010.
- [8] C. Kingkan, S. Ito, S. Arai, T. Nammoto, and K. Hashimoto, "Model-based virtual visual servoing with point cloud data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 5549–5555.
- [9] J. Kinugawa, A. Kanazawa, S. Arai, and K. Kosuge, "Adaptive task scheduling for an assembly task coworker robot based on incremental learning of human's motion patterns," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 856–863, Apr. 2017.
- [10] D. Liu, J. Kinugawa, and K. Kosuge, "A projection-based making-human-feel-safe system for human-robot cooperation," in *Proc. IEEE Int. Conf. Mechatronics Autom.*, Aug. 2016, pp. 1101–1106.
- [11] S. Arai, A. L. Pettersson, and K. Hashimoto, "Fast prediction of a Worker's reaching motion without a skeleton model (F-PREMO)," *IEEE Access*, vol. 8, pp. 90340–90350, 2020.
- [12] Y. Xu, S. Arai, F. Tokuda, and K. Kosuge, "A convolutional neural network for point cloud instance segmentation in cluttered scene trained by synthetic data without color," *IEEE Access*, vol. 8, pp. 70262–70269, 2020.
- [13] D. Li, H. Wang, N. Liu, X. Wang, and J. Xu, "3D object recognition and pose estimation from point cloud using stably observed point pair feature," *IEEE Access*, vol. 8, pp. 44335–44345, 2020.
- [14] A. Aldoma, M. Vincze, N. Blodow, D. Gossow, S. Gedikli, R. B. Rusu, and G. Bradski, "CAD-model recognition and 6DOF pose estimation using 3D cues," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCV Workshops)*, Nov. 2011, pp. 585–592.
- [15] K. Harada, W. Wan, T. Tsuji, K. Kikuchi, K. Nagata, and H. Onda, "Iterative visual recognition for learning based randomized bin-picking," in *Proc. ISER*, Tokyo, Japan, 2016, pp. 646–655.
- [16] A. Aldoma, F. Tombari, R. B. Rusu, and M. Vincze, "OUR-CVFH oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6DOF pose estimation," in *Proc. Joint DAGM (German Assoc. Pattern Recognit.) OAGM Symp.*, Berlin, German, 2012, pp. 112–122.

- [17] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, "Aligning point cloud views using persistent feature histograms," in *Proc. IROS*, Nice, France, 2008, pp. 3384–3391.
- [18] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *Proc. ICRA*, Kobe, Japan, 2009, pp. 3212–3217.
- [19] S. Salti, F. Tombari, and L. Di Stefano, "SHOT: Unique signatures of histograms for surface and texture description," *Comput. Vis. Image Understand.*, vol. 125, pp. 251–264, Aug. 2014.
- [20] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3D scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 5, pp. 433–449, May 1999.
- [21] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3D object recognition," in *Proc. CVPR*, San Francisco, CA, USA, 2010, pp. 998–1005.
- [22] M. Mohamad, D. Rappaport, and M. Greenspan, "Generalized 4-points congruent sets for 3D registration," in *Proc. Int. Conf. 3D Vis*, Lyon, France, 2014, pp. 83–90.
- [23] O. Tuzel, M. Y. Liu, Y. Taguchi, and A. Raghunathan, "Learning to rank 3D features," in *Proc. ECCV*, Zurich, Switzerland, 2014, pp. 520–535.
- [24] S. Arai, N. Fukuchi, and K. Hashimoto, "FAst detection algorithm for 3D keypoints (FADA-3K)," *IEEE Access*, vol. 8, pp. 189556–189564, 2020.
- [25] L. Kiforenko, B. Drost, F. Tombari, N. Krüger, and A. G. Buch, "A performance evaluation of point pair features," *Comput. Vis. Image Understand.*, vol. 166, pp. 66–80, Jan. 2018.
- [26] S. Hinterstoisser, S. Holzer, C. Cagniard, S. Ilic, K. Konolige, N. Navab, and V. Lepetit, "Multimodal templates for real-time detection of textureless objects in heavily cluttered scenes," in *Proc. ICCV*, Barcelona, Spain, 2011, pp. 858–865.
- [27] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of textureless 3D objects in heavily cluttered scenes," in *Proc. ICCV*, Daejeon, South Korea, 2012, pp. 548–562.
- [28] R. Rios-Cabrera and T. Tuytelaars, "Discriminatively trained templates for 3D object detection: A real time scalable approach," in *Proc. ICCV*, Sydney, NSW, Australia, 2013, pp. 2048–2055.
- [29] R. He, J. Rojas, and Y. Guan, "A 3D object detection and pose estimation pipeline using RGB-D images," in *Proc. ROBIO*, Macau, China, 2017, pp. 1527–1532.
- [30] L. Bo, X. Ren, and D. Fox, "Unsupervised feature learning for RGB-D based object recognition," *Exp. Robot.*, vol. 88, pp. 387–402, Jan. 2013.
- [31] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," 2017, *arXiv:1711.00199*. [Online]. Available: <http://arxiv.org/abs/1711.00199>
- [32] J. M. Wong, V. Kee, T. Le, S. Wagner, G.L. Mariottini, A. Schneider, L. Hamilton, R. Chipalkatty, M. Hebert, D. M. S. Johnson, J. Wu, B. Zhou, and A. Torralba, "SegICP: Integrated deep semantic segmentation and pose estimation," in *Proc. IROS*, Vancouver, BC, Canada, 2017, pp. 5784–5789.
- [33] D. Liu, S. Arai, J. Miao, J. Kinugawa, Z. Wang, and K. Kosuge, "Point pair feature-based pose estimation with multiple edge appearance models (PPF-MEAM) for robotic bin picking," *Sensors*, vol. 18, no. 8, p. 2719, Aug. 2018.
- [34] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. ICCV*, Venice, Italy, 2017, pp. 2980–2988.
- [35] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.
- [36] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. ICCV*, Santiago, Chile, 2015, pp. 1026–1034.
- [37] J. Redmon, S. R. Divvala, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. CVPR*, Las Vegas, NV, USA, 2016, pp. 779–888.
- [38] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. ECCV*, Amsterdam, The Netherlands, 2016, pp. 21–37.
- [39] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. ECCV*, Venice, Italy, 2017, pp. 7263–7271.
- [40] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "DSSD: Deconvolutional single shot detector," 2017, *arXiv:1701.06659*. [Online]. Available: <http://arxiv.org/abs/1701.06659>
- [41] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Adv. Neural Inf. Process Syst.*, Barcelona, Spain, 2016, pp. 379–387.
- [42] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. CVPR*, Columbus, OH, USA, 2014, pp. 580–587.
- [43] R. Girshick, "Fast R-CNN," in *Proc. ICCV*, Santiago, Chile, 2015, pp. 1440–1448.
- [44] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process Syst.*, Montreal, QC, Canada, 2015, pp. 91–99.
- [45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, Las Vegas, NV, USA, 2016, pp. 770–778.
- [46] T. Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," in *Proc. CVPR*, San Francisco, CA, USA, 2017, pp. 2117–2125.
- [47] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Proc. ECCV*, Zurich, Switzerland, 2014, pp. 740–755.
- [48] D. Liu, S. Arai, Z. Feng, J. Miao, Y. Xu, J. Kinugawa, and K. Kosuge, "2D object localization based point pair feature for pose estimation," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, Dec. 2018, pp. 1119–1124.
- [49] D. Liu, S. Arai, F. Tokuda, Y. Xu, J. Kinugawa, and K. Kosuge, "Deep-learning based robust edge detection for point pair feature-based pose estimation with multiple edge appearance models," in *Proc. ROBIO*, Dali, China, 2019, pp. 2920–2925.
- [50] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *Proc. ICRA*, Shanghai, China, 2011, pp. 1–4.
- [51] M. Li and K. Hashimoto, "Curve set feature-based robust and fast pose estimation algorithm," *Sensors*, vol. 17, no. 8, p. 1782, Aug. 2017.
- [52] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, Feb. 1992.



DIYI LIU received the M.S. and Ph.D. degrees in engineering from Tohoku University, Sendai, Japan, in 2016 and 2019, respectively.



SHOGO ARAI (Member, IEEE) received the B.S. degree in aerospace engineering and M.S. and Ph.D. degrees in information sciences from Tohoku University, Sendai, Japan, in 2005, 2007, and 2010, respectively.

From 2010 to 2016, he was an Assistant Professor with the Intelligent Control Systems Laboratory, Tohoku University. He joined the System Robotics Laboratory, Department of Robotics, Tohoku University, as an Associate Professor, in 2016, where he is currently an Associate Professor. His research interests include robot vision, machine vision, 3D measurement, production robotics, networked control systems, and multi-agent systems. He received the Best Paper Award from FA Foundation, in 2019, the 32th Best Paper Award from the Robotics Society of Japan, in 2019, the Certificate of Merit for Best Presentation from the Japan Society of Mechanical Engineers, in 2019, the Excellent Paper Award from the Institute of Systems from Control and Information Engineers, in 2010, the Best Paper Award Finalist at IEEE International Conference on Mechatronics and Automation, in 2012, the SI2019 Excellent Presentation Award from the Society of Instrument and Control Engineers, in 2019, the SI2018 Excellent Presentation Award from the Society of Instrument and Control Engineers, in 2018, the SI2017 Excellent Presentation Award from the Society of Instrument and Control Engineers, in 2017, and the Graduate School Research Award from the Society of Automotive Engineers of Japan, Inc., in 2007.



YAJUN XU received the B.S. degree in mechanical engineering from Central South University, Changsha, China, in 2015, and the M.E. degree from the Graduate School of Engineering, Tohoku University, Japan.

He is currently a student with the Graduate School of Information Science and Technology, Hokkaido University, Japan. His research interests include deep learning, 2D/3D segmentation, and 6D pose estimation.



FUYUKI TOKUDA (Member, IEEE) received the B.S. degree in engineering from the Nagoya Institute of Technology, Nagoya, Japan, in 2017, and the M.S. degree in engineering from Tohoku University, Sendai, Japan, in 2019, where he is currently pursuing the Ph.D. degree.

His research interest includes development of visual feedback control using deep learning for bin-picking and assembling of industrial parts.



KAZUHIRO KOSUGE (Fellow, IEEE) received the B.S., M.S., and Ph.D. degrees in control engineering from the Tokyo Institute of Technology, in 1978, 1980, and 1988, respectively.

From 1980 to 1982, he was a Research Staff with the Production Engineering Department, Nippon Denso Company Ltd. (DENSO Company Ltd.). From 1982 to 1990, he was a Research Associate with the Department of Control Engineering, Tokyo Institute of Technology. From 1990 to 1995,

he was an Associate Professor with Nagoya University. Since 1995, he has been with Tohoku University, Japan, where he is currently a Professor with the Department of Robotics.

Dr. Kosuge is a JSME Fellow, SICE Fellow, RSJ Fellow, and a JSAE Fellow. He is a member of the IEEE-Eta Kappa Nu. He received the JSME Awards for the best papers from the Japan Society of Mechanical Engineers, in 2002 and 2005, and the RSJ Award for the best papers from the Robotics Society of Japan, in 2005. He was a President of the IEEE Robotics and Automation Society, from 2010 to 2011, and the IEEE Division X Director, from 2015 to 2016. He was the IEEE Vice President for Technical Activities, in 2019.

...