Received 28 October 2022, accepted 16 November 2022, date of publication 21 November 2022, date of current version 28 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3223695

RESEARCH ARTICLE

Object Pose Estimation and Feature Extraction Based on PVNet

YI-HSIANG KAO[1], CHING-KUN CHEN[1], CHIH-CHENG CHEN[1], (Senior Member, IEEE), AND CHEN-YEN LAN[2]
[1]Department of Automatic Control Engineering, Feng Chia University, Taichung 40724, Taiwan
[2]Node Health Technology Company Ltd., Taichung 042708, Taiwan

Corresponding author: Ching-Kun Chen (chingkchen@fcu.edu.tw)

This work was supported by the Ministry of Science and Technology, Taiwan, under Grant MOST 110-2221-E-035-087.

ABSTRACT In traditional industrial fields, a robot arm is usually used for high-precision or highly repetitive movements, but now, with the development of three-dimensional (3D) stereo machine vision in smart manufacturing, the smart factory has moved toward the development of a robot arm combined with the image recognition technology. Currently, in the manufacturing industry, most of the images for computing are obtained using two-dimensional (2D) machine vision; here, the 2D advantage is that the camera lens can obtain the simulation of the plane color pixel, but the disadvantage is that it cannot obtain the real space depth distance information, resulting in a more accurate analysis of the workpiece position and features. Therefore, in this study, a pixel-wise voting network (PVNet)-based object pose estimation and feature extraction was developed to perform more diverse object testing for the considered network model. Unlike other workpiece picking systems for smart manufacturing, most of the systems today still framed the workpiece in two dimensions only, but the approach proposed in this paper framed the workpiece pose in three dimensions. Thus, the network could successfully predict the pose even when the workpiece was obscured or the image was not fully captured. The images were input into the neural network by means of supervised learning, and training was performed using transformation matrices between multi-angle images of the artifacts and feature points extracted from the 3D models. The results of this study revealed the pose estimation results of various objects at different viewing angles and proposed the feature gripping strategy for the robot arm to follow this process in the future.

INDEX TERMS 3D stereo machine vision, smart manufacturing, pose estimation, feature extraction.

I. INTRODUCTION

With the rise of Industry 4.0, it is all about smart production, which means using various integrated sensor systems to combine a human-centered approach into more flexible manufacturing, and using technologies such as information technology, cloud computing, and business analytics to achieve the goal of rapid adjustment and strategy setting [1], [2]. Thus, the term ‘‘smart factory'' was coined.

There are already many applications in flexible production-oriented smart factories that have introduced robotic arms as automated logistics or processing equipment, with the identification of diverse workpieces as the main intelligent challenge and three-dimensional (3D) visual recognition as the mainstream development solution [3], [4]. In recent years, the increasing labor costs and health consciousness have led to the rapid growth of industrial robot usage and market, and robots need to become more flexible and intelligent if they are to replace or assist operators in the related tasks. In addition, with the increasing precision of the various parts produced by various companies, the task of detecting scratches that are difficult to notice by the naked eye requires industrial-grade machine vision systems. Chamberlain et al. [5] published a report on the application of machine vision systems based on their rapid processing ability to solve the labor-intensive quality inspection and analysis of the causes of quality defects,

The associate editor coordinating the review of this manuscript and approving it for publication was Jon Atli Benediktsson.

when the production line of the operating robots into the vision-assisted systems equips the production robots with more complex and sophisticated operational capabilities, such as the application of automotive production lines and pharmaceuticals. The application of vision-assisted systems in the production line will enable the robots to have more complex and precise operation, such as in the automotive production line, pharmaceuticals, packaging of various products, and food industry.

In the past, in traditional industrial fields, in the case of stacked workpieces, to make the clamping of the robot arm on the production line convenient, almost all of them used vibrations or shunts to disperse the workpieces in advance, and once the workpieces were large or heavy, they used visual methods to distinguish the position and the direction of the workpieces along with certain other information. Therefore, the traditional robotic arm technology can only pick up ''clear and well-defined workpieces,'' and if a bunch of irregular objects are placed in the same box, it is impossible to pick them up smoothly. Therefore, random bin picking is still a major issue in smart manufacturing. In the traditional problem of determining the 3D pose of a workpiece, the common method used is the (Perspective-n-Point, PnP) algorithm [6], which aims to solve the changes in the two-dimensional (2D) points corresponding to the 3D points, and can be recognized as the estimation of the camera pose under the condition that the coordinates of N 3D spatial points and their projection in the 2D image position are known. The objective of the present study was to extract the characteristic points of a workpiece and estimate its 6D pose through the regression image coordinates before the robot arm clamped the workpiece. However, because RGB-D cameras have limitations in terms of the frame rate, field of view, and depth range, their effects are affected to a certain extent to improve their results under occlusion.

The network architecture used in this study was trained by a modified network model using ResNet18 as the backbone, as proposed by Peng et al. [7]. The deep convolutional neural network technique for target detection on RGB images has been well demonstrated and has been used to improve six-dimensional (6D) target pose detection [8], [9], [10]. Therefore, to improve the recognition efficiency on the production line, in this study, firstly, in the prior work we tested the feasibility of connecting the 3D camera to the computer and ensuring its stability, and later on, the training images of various objects were acquired by means of a handheld camera. After collecting the images of the objects, various image information is processed by semi-automatic methods, such as point cloud reconstruction, segmentation…etc. Finally, the object point cloud model is recovered and its 3D feature points are extracted from it. Finally, the PVNet network model is trained to perform the estimation task of the artifact pose.

The image quality trained by neural network is considered as an important information. We learned from the actual 3D camera shots that the trained estimation results are poor when the images are blurred. In this study, we not only tested the pose estimation under the LINEMOD dataset from previous studies [7], but also conducted a new dataset with the hardware equipment used in the proposed system, and determined the significance of the representation in the transformation matrix.

II. SYSTEM DESCRIPTION AND METHODS

The flowchart of this research is shown in Fig. 1. First, a 3D camera (Intel RealSense) was used to take the pictures of object. At the same time, through the characteristics of USB connection with the computer, we can obtain colored images (RGB), internal parameters of the camera, and depth map information. From the RGB image and camera internal parameters, we could obtain the pose matrix of each viewpoint map. Meanwhile, to obtain the pose information of each viewpoint, we used an ArUco tag for pose orientation to obtain the conversion matrix between images, and through this conversion matrix [11], [12], we aligned the point clouds generated from each angle and then generated a complete scene point cloud. The data of the point cloud not only provided the RGB color information but also helped to determine the coordinates of the workpiece in space.

Then, we segmented the generated scene point cloud by using the MeshLab tool for object segmentation and determined the 3D feature points from the segmented artifact point cloud by using the farthest point sampling method. Eight feature points were calculated from the center of the artifact point cloud. The 3D feature points were aligned with the 2D feature points by using the pose matrix, and the information of these marker points was used as the Pixel-wise Voting Network (PVNet) input; the final output through the PVNet network provided the results of the 6D pose boundary frame with different angles and the error of the real marker data.

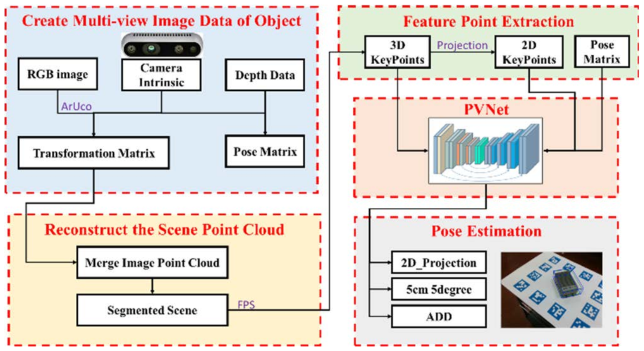

FIGURE 1. Schematic illustration of object pose estimation and feature extraction.

A. PERSPECTIVE PROJECTION METHOD

The 6D workpiece pose was one of the most important pieces of information obtained in this study. The correct pose could accurately guide the robot arm in the correct direction for gripping tasks. However, the RGB image generated by the camera could only provide 2D information, so the depth

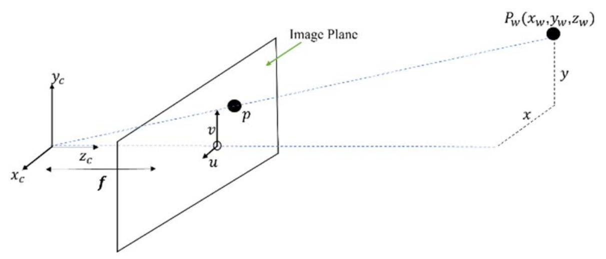

FIGURE 2. Schematic illustration of object pose estimation and feature extraction.

map and 3D point cloud were used to know the location of the workpiece feature points. Furthermore, through perspective projection, a camera projection matrix of size 3∗4 was calculated, which implied that the 3D feature points of the workpiece corresponded to the two-dimensional points on the RGB image. This method not only helped us to calculate the workpiece pose but also presented the result of the 6D pose on the RGB image in a more intuitive manner.

1) CAMERA MODEL

Figure 2 shows the model of the camera in perspective projection. The 3D point P_w in the camera coordinate system was transformed to correspond to the 2D point p in the imaging plane in the image coordinate system; such a transformation process could also be called projective transformation, where the focal length of the camera could be calculated assuming that the coordinates of the 3D point P_w in space under the camera coordinate system were $P_w = [x_w, y_w, z_w]$ and the coordinates in the image coordinate system were $p = [u, v]$. Thus, on the basis of the principle of similar triangles:

$$\frac{x_w}{u} = \frac{z_w}{f} = \frac{y_w}{v} \tag{1}$$

where x_w, and y_w, and z_w are the coordinate points of P_w in the world coordinate system. Furthermore, the coordinates of point p in the image plane could be derived as follows:

$$u = f\frac{x_w}{z_w}, \quad v = f\frac{y_w}{z_w} \tag{2}$$

where u and v are the coordinate points of p in the image plane. Expressed in the matrix form,

$$z_w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \tag{3}$$

Through the above conversion formula, we projected the camera coordinate system onto the image coordinate system, which also implied transferring the 3D information to the 2D plane. Finally, we used the bilinear interpolation method to convert the origin of the image coordinate system uv into the origin of the pixel coordinate system xy, and the geometric relationship was obtained as follows:

$$\begin{cases} u = \dfrac{x}{d_x} + c_x \\ v = \dfrac{y}{d_y} + c_y \end{cases} \tag{4}$$

where d_x denotes the direction width of pixel x, d_y indicates the direction width of pixel y, and c_x, c_y represent the coordinate point on the screen.

The representation in the matrix form would be as follows:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{d_x} & 0 & c_x \\ 0 & \frac{1}{d_y} & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{5}$$

One of the important parameters of the camera is the internal parameter K. This internal parameter is composed of two parts. One is the projective transformation, which denotes the distance from the focus of the camera to the imaging plane, that is, the focal length in (1). The other is the coordinates (Focal point) of the main point (c_x, c_y) projected on the imaging plane. Therefore, we combined the matrices of (3) and (5) to obtain the following intrinsic parameter matrix K:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

In this study, the internal parameters were used in the depth camera of RealSense, and thus, they could be obtained by the official function called get_intrinsics() [13]. Furthermore, we expressed the internal parameter matrix of the camera when capturing UNO-2272G of Case1 as follows:

$$K = \begin{bmatrix} 610.897827 & 0 & 327.844085 \\ 0 & 610.898620 & 237.670547 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

2) PERSPECTIVE PROJECTION

Perspective projection transformation is aimed at transferring the world coordinate system to the camera coordinate system, also known as rigid transformation [14]. In this study, we rotated, translated, and scaled the original world coordinate system, and then projected the camera coordinate system onto the image coordinate system first and then through discrete sampling, onto the pixel coordinate system. Thus, we found the best rotation matrix R^{3*3} and translation vector t^{3*1} by using the external parameters of the camera in the transformation of 2D RGB image and 3D space, and the internal value of the external parameter matrix T could be expressed as follows:

$$T = \begin{bmatrix} R^{3*3} & t^{3*1} \end{bmatrix} \tag{8}$$

In the world coordinate system, we assumed that point X in space could be defined and that its origin position plus the direction of the 3D axes was equal to the attitude of the coordinate system. Therefore, in the definition of the camera coordinate system, the origin was the center of the camera, and thus, the representative point X_w was expressed as $X_w = [x_w, y_w, z_w]^T$ in the world coordinate system and $X_c = [x_c, y_c, z_c]^T$ in the camera coordinate system. As mentioned earlier, the main task was to first scale down the world coordinates and add rotation and translation to rotate around the z-axis, y-axis, and x-axis, as represented in (9), (10), and (11).

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = R_1 \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} \tag{9}$$

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = R_2 \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} \tag{10}$$

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = R_3 \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} \tag{11}$$

Finally, these coordinates were transformed into the camera coordinates, according to the complete transformation equation (12), and the internal parameters of the camera model were obtained as follows. Combining these with the conversion matrix of (12), we obtained the complete operation shown in (13).

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} R^{3*3} & t^{3*1} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{12}$$

$$\begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R^{3*3} & T^{3*1} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{13}$$

B. DATASET GENERATION AND POSE LOCATION

In order to get a more realistic image of the actual workpiece on the production line, the RealSense D435i depth camera was used to take pictures of the scene. In addition, the ArUco marker data were used to surround the workpiece to be photographed as the current camera pose estimation, and the transformation matrix between view angles was stored as an auxiliary tool to create the complete point cloud data. Through the depth camera of RealSense, the RGB image, depth map, and internal parameters of the camera were acquired. First, the workpiece to be photographed was placed in the established field, and the camera position was not fixed. Furthermore, to ensure the accuracy of the rotational attitude, at least two ArUco markers were included in the image during the shooting process to facilitate the calculation of the rotational attitude matrix between images.

1) ARUCO

ArUco is a camera pose estimation library embedded in OpenCV [15]. The image is similar to a binary barcode QRCode, and each individual ArUco has its own specification and ID number. Furthermore, it is composed of a dark border

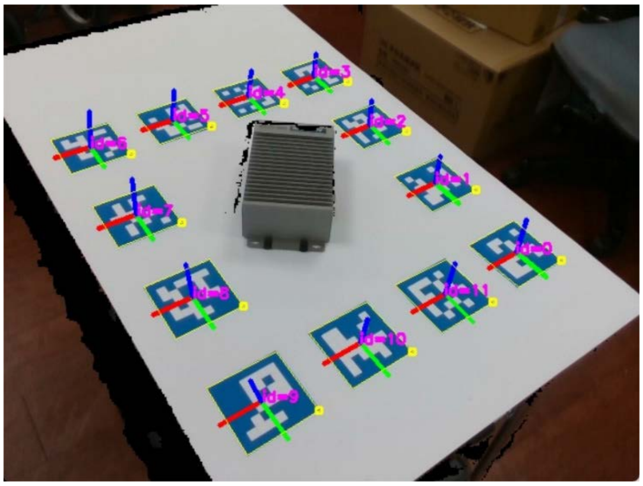

FIGURE 3. ArUco pose estimation and marking results.

and a square composite marker structure with an internal binary matrix. The size of the label determines the size of the internal matrix, e.g., a 4 × 4 ArUco is composed of 16 bits. In this study, we created 12 different types of ArUco of size 6 × 6 and generated the marker images by using the drawMarker() function, which contains five parameters. The second parameter was the marker ID, so the valid ID values were 0–249 according to the ArUco used in this study, and the subsequent parameters were the output image size, the output image, and the specified border width.

2) CAMERA POSE LOCATION

Even without the original CAD file of the workpiece, we could restore the point cloud of the workpiece with the RGB image, depth map, and the internal parameters of the camera with the depth camera. The inspection process was mainly based on segmentation and contour extraction with adaptive thresholds for the markers, and then, the markers were identified by analyzing the internal codes. In order to accurately restore the workpiece point clouds, we computed the pose relationship between the viewpoint images, so the important task was to obtain the camera pose after detecting the ArUco marker. While the pose estimation was similar to the important pose estimation in this study, the difference was the target coordinate system for the conversion carried out mainly through a PnP algorithm to solve the coordinate system conversion problem from 3D to 2D. However, the ArUco module also provides a function for estimating the marker pose. Therefore, we used this function to estimatePoseSingleMarkers() to detect the camera pose, and the estimation result is shown in Fig. 3.

3) POINT CLOUD REGISTRATION

In order to reconstruct the point clouds of the scene, we found the optimal rotation and translation matrix for different viewpoint point clouds based on singular value decomposition (SVD) [16] and then, aligned the point clouds corresponding to each viewpoint map [17]. First, we determined the center

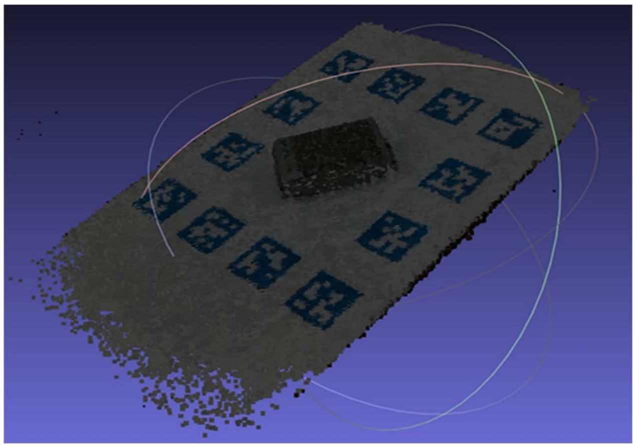
FIGURE 4. Scene point cloud merged result.

of mass of the two different point cloud datasets; it was calculated as follows:

$$\left\{\begin{array}{l} centroid_A = \frac{1}{N}\sum_{i=1}^{N} P_A^i \\ centroid_B = \frac{1}{N}\sum_{i=1}^{N} P_B^i \end{array}\right\} \tag{14}$$

where P_A and P_B are assumed as the points in the two different point cloud datasets.

Then, to find the optimal rotational matrix, the center of mass of the two datasets was positioned at the same origin, and a matrix of M was obtained by simply calculating the rotation angle as in (15). Next, SVD was applied to matrix M, and the rotational matrix R was obtained on the basis of matrix U and matrix V. The result was as follows:

$$M = \sum_{i=1}^{N}\left(P_A^i - centroid_A\right)\left(P_B^i - centroid_B\right)^T \tag{15}$$

$$\left\{\begin{array}{c} [U, S, V] = SVD(M) \\ R = VU^T \end{array}\right\} \tag{16}$$

Finally, we combined the rotating matrix R and the center of mass to obtain a 4×4 transformation matrix T. After obtaining the transformation matrix between the viewpoints, we determined the overlapping parts of the point clouds by using the KD-tree nearest neighbor search. Because of the large number of point clouds in the 3D space, the KD-tree method reduced the total time consumption, ensured the search of associated points between point clouds, and maintained the real time alignment. The corresponding result is shown in Fig. 4.

4) WORKPIECE POINT CLOUD DIVISION

For the point cloud segmentation [18], [19], we used the MeshLab tool suite to manually remove the background point clouds and reconstruct the mesh of the workpiece. First, we used the Select Vertice tool to remove the point clouds outside of the workpiece body; the corresponding result is shown in Fig. 5.

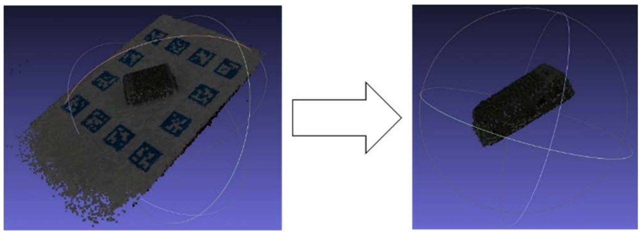
FIGURE 5. Select vertice diagram.

The normal vector is an indispensable piece of information in the processing of 3D point clouds because of its important locality property. The point cloud obtained by the depth camera only records the coordinates of each point in a 3D space, because there is no meaning of the normal vector and therefore no connection between points. We also calculated the normal vector of the point cloud in the 3D space by using Compute for the point set.

Second, after calculating the normal vector, we reconstructed the surface of the workpiece; the corresponding result is shown in Fig. 6. The main principle of this method is that the point cloud represents the position of the surface of the workpiece [20], and the normal vector representing the inner and outer directions is used to derive the smooth surface of the workpiece by using the indicator function χ_M produced by the workpiece, which is defined as follows:

$$\chi_M(x) = \left\{\begin{array}{l} 1x \in M \\ 0x \in M \end{array}\right\} \tag{17}$$

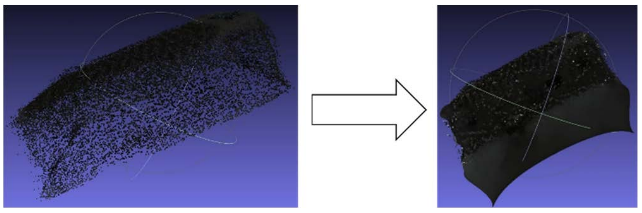
FIGURE 6. Poisson surface reconstruction results.

Finally, we removed the excess parts from the reconstructed result to obtain the remaining parts where the real workpiece belonged to, and then, filled the empty parts by using the Close Holes tool, as shown in Fig. 7.

5) FARTHEST POINT SAMPLING

To establish the bounding box of the object, the extraction of 3D feature points was particularly important. Moreover, the 3D feature points were used to calculate the correspondence between the coordinate points on the 2D plane by using the projection transformation matrix used in this study. The farthest point sampling method (FPS) was adopted [21], because this method can be uniformly sampled for point cloud data and is widely used. Moreover, the main goal of this algorithm is to select the farthest point. The farthest feature point is extracted and voted to calculate the attitude.

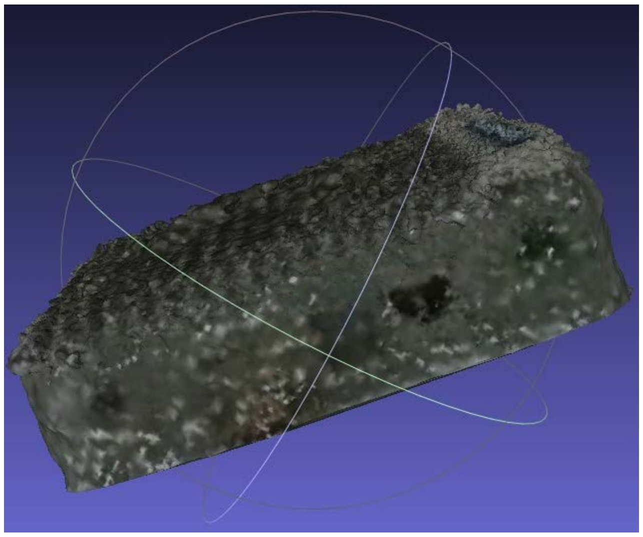
FIGURE 7. Final workpiece reconstruction result.

As shown in Fig. 8, FPS can be mainly divided into three steps.

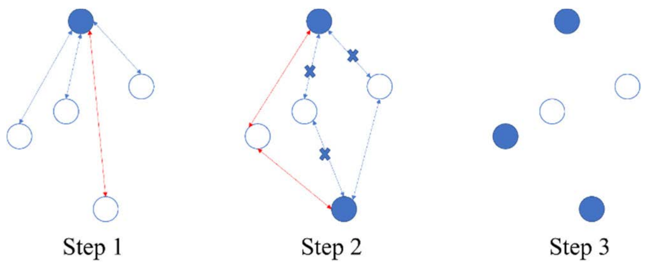

FIGURE 8. Principle of the farthest point sampling algorithm.

The first step is to input the point cloud data for calculation and to arbitrarily select the initial point. In this study, the object was selected. The center point is the initial point. The second step is to select the point farthest from the initial point as the second point. The third step is to calculate the point farthest from the initial point and the second point as the third point. Then, iterate these steps until the number of feature points reaches the set number.

III. FEATUR EXTRACTION AND POSE ESTIMATION

In this section, we introduce the PVNet neural network based on the ResNet18 architecture for feature point extraction and pose estimation. As shown in Fig. 9, first, we took a tag file containing RGB images, mask images, internal camera parameters, feature points, 2D and 3D projection conversion matrix, etc. as the input data for the neural network. The pose matrix was generated by transforming the object space coordinate system to the camera coordinate system. In the training phase, we learnt the network model from the real ground-truth data, so that when a new view angle image was input to the PVNet network for prediction, the pose of the artifact in the new image could be estimated quickly by learning the training results.

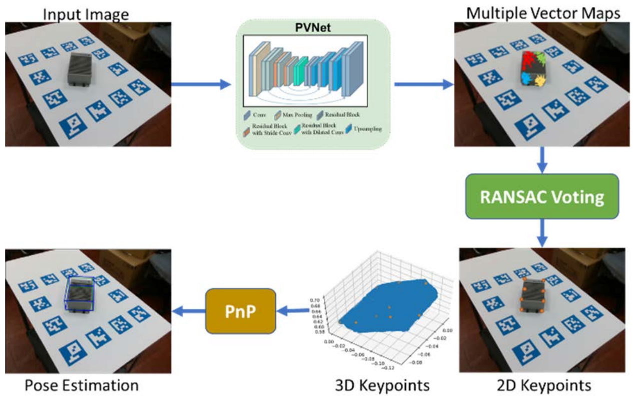

FIGURE 9. Feature point extraction and pose estimation based on PVNet.

A. FEATURE POINT SELECTION

In the past, some algorithms used a convolution neural network (CNN) to return 2D feature points and then calculated their 6D pose with the PnP algorithm. However, the feature point extraction method used in this study used the FPS algorithm to calculate the 3D feature points of the ground truth. Thereafter, the 3D to 2D transformation matrix was obtained by using the perspective projection method to determine the 3D feature points and the corresponding 2D feature points. However, in the prediction stage, a pixel-level voting neural network (PVNet) was used to predict the 2D feature points in an analogous search manner, which could present the pose estimation results without affecting the blocked or truncated images.

In order to reduce the localization error of the feature points, it was necessary to perform feature point extraction based on the 3D point cloud data of the workpiece instead of the eight vertices of the bounding box. If the eight vertices of the bounding box were used as the feature points, when the coordinates of each vertex were far away from the target object, the positioning error would have been larger as the distance from the target pixel increased. The logic of the FPS algorithm is to assume that there are N points and all points are classified into two different sets A and B. Set A represents the set of selected points, and Set B represents the set of unselected points. With this principle, first, we set the center of the workpiece as the initial point and then calculated the farthest point from the initial point. We repeatedly searched for the feature points on the surface of the workpiece and categorized the points farthest from each current feature point into the selected set.

B. PVNET NETWORK ARCHITECTURE

This architecture is modified from the model architecture of ResNet18 and is divided into three parts. First, RGB images (size: 640×480) with ArUco markers are input to the pixel-based voting neural network to obtain a feature map of the same size as the input image. Strictly speaking, the 6D pose is the rigid transform from the world coordinate to the camera coordinate, and the result of the pose estimation is plotted in the output image as a 3D bounding box.

1) RESNET18 REFINED NEURAL NETWORK

We used ResNet18 as the backbone of PVNet to predict the location of the feature points of the artifacts in the image. The network architecture mainly consisted of a series of convolution layer, pooling layer, upper sampling layer, and residual block with stride convolution and dilated convolution. The residual block was composed of the upper sampling layer with stride convolution and dilated convolution. The difference from the original ResNet18 network was that we first took an original image of size 640 × 480 × 3 as the input image and performed down-sampling with the stride volume set as 2 until the image feature map size was 80 × 60. Then, we discarded the subsequent pooling layer and did not perform further down-sampling. At the same time, to avoid the omission of some feature information as in the case of traditional pooling, the same feature information was retained without reducing the image size, by expanding the convolution feature during the process. Finally, the fully connect layer in the original architecture was replaced with a convolution layer, and the previously obtained feature map was repeatedly skip connected, convolved, and up-sampled to return the feature map to its original size of 640 × 480. The final convolution layer yielded the unit vector $\boldsymbol{V_k(P)}$ of each pixel and the class label, which represents the semantics of a specific object; $\boldsymbol{V_k(P)}$ represents the direction of each pixel P towards each 2D feature $\boldsymbol{X_k}$. The definition of the unit vector $\boldsymbol{V_k(P)}$ is as follows:

$$V_k(P) = \frac{X_k - P}{\|X_k - P\|_2} \tag{18}$$

Given the segmentation label and unit vectors, in this study, we formulated the hypothesis of feature points on the basis of the RANSAC voting mechanism [22], extracted the pixel range of the target artifact from the class label, randomly selected two pixels, and used the intersection of their vectors as the hypothesis $\boldsymbol{h_{k,i}}$ of 2D feature points $\boldsymbol{x_k}$. The set of feature point positions $\{\boldsymbol{h_{k,i}} | \boldsymbol{i} = \boldsymbol{1, 2, \ldots, N}\}$ was generated by repeating N times, and the voting scores of these hypothetical points are defined as $\boldsymbol{w_{k,i}}$ in (19), where II represents the pointer function, θ is set to a threshold value of 0.99, and $\boldsymbol{P} \in \boldsymbol{O}$ represents the pixel P belonging to the target workpiece $\boldsymbol{O}$.

$$w_{k,i} = \sum_{P \epsilon O} \mathbb{I}\left(\frac{(h_{k,i} - P)^T}{\|h_{k,i} - P\|_2} V_k(P) \geq \theta\right) \tag{19}$$

2) 6D POSE ESTIMATION

We can solve it by using the PnP algorithm, but in the traditional PnP algorithm, there is uncertainty and variability in the confidence of the feature points, and the probability distribution of each feature point is obtained by the voting system. The final evaluation criterion for each feature is obtained by calculating the mean $\boldsymbol{\mu_k}$ as in (20):

$$\mu_k = \frac{\sum_{i=1}^{N} w_{k,i} h_{k,i}}{\sum_{i=1}^{N} w_{k,i}} \tag{20}$$

and the covariance matrix $\sum_{\boldsymbol{k}}$ is expressed as (21), where $\boldsymbol{k = 1, 2, \ldots, K}$.

$$\Sigma_k = \frac{\sum_{i=1}^{N} w_{k,i} \left(h_{k,i} - \mu_k\right)\left(h_{k,i} - \mu_k\right)^T}{\sum_{i=1}^{N} w_{k,i}} \tag{21}$$

The 6D pose was obtained by minimizing the Mahalanobis distance [23], as defined in (22), where Z_k denotes the 3D feature coordinate and $\tilde{z}_k$ represents the corresponding point of the 3D feature projected on 2D.

$$\left\{\begin{array}{c} \tilde{z}_k = \pi\left(RZ_k + t\right) \\ minimize_{R,T} \sum_{k=1}^{K} \left(\tilde{z}_k - \mu_k\right)^T \sum_k^{-1} \left(\tilde{z}_k - \mu_k\right) \end{array}\right\} \tag{22}$$

We also used this pose estimation result to calculate different evaluation metrics, such as the 2D projection error and the 3D model point average distance (ADD) of asymmetric objects [24]. If the result was less than five pixels, then the pose estimation was correct. In this study, the threshold value was set to the diameter of the workpiece pixels and the five-pixel diameter distance was calculated proportionally. The two calculations are shown in (23) and (24), respectively. In addition, the pose estimation results were evaluated using the 5 cm-5° index. When the rotation angle error was less than 5° and the translation error was less than 5 cm, the pose estimation was correct.

$$2D_{Projection} = \frac{1}{m}\sum_{x \in M}\left\|K\left(R_x + t\right) - K\left(\hat{R}_X + \hat{t}\right)\right\| \tag{23}$$

$$ADD = \frac{1}{m}\sum_{x \in M}\left\|\left(R_x + t\right) - \left(\hat{R}_X + \hat{t}\right)\right\| \tag{24}$$

3) LOSS FUNCTION

In deep learning, the selection of the loss function is necessary, and the common loss functions can be divided into two major categories. One is the least absolute error, also known as L1 Loss, and the other is the least square error, also known as L2 Loss. The drawback of L1 Loss is that it is not smooth at the zero point, while L2 Loss is easily affected by outliers. To improve these shortcomings, Smooth L1 Loss was created. Moreover, to have more stable robustness under various data, in this study, we adopted Smooth L1 Loss to learn the unit vector, and its loss function was as defined in (25) and (26).

$$\triangle v_k(P; w) = \tilde{v}_k(P; w) - v_k(P) \tag{25}$$

$$\ell(w) = \sum_{k=1}^{K}\sum_{P \in O} \ell_1\left(\triangle v_k(P; w)|_x\right) + \ell_1\left(\triangle v_k(P; w)|_y\right) \tag{26}$$

where $\triangle v_k|_x$ and $\triangle v_k|_y$ represent two different elements of $\triangle v_k$ and w represents the parameter in the PVNet network, v_k is the base unit vector, and $\tilde{v}_k$ is the prediction vector.

In the training process, the segmentation loss and the voting loss were also calculated, in which the CrossEntropy loss function was used to determine the closeness of the real output to the desired output. Therefore, when the classification training was performed, the output node of a class had to be 1 and the output of the other nodes had to be 0; this resulted in a set of labels for this class, which was the desired output of the neural network and measured the difference between the output of the network and the class labels.

IV. EXPERIMENTS RESULTS

In this study, we tested a variety of workpieces by using a depth camera to capture multiple RGB images (size: 640 × 480) at 40-s intervals around the workpiece as the network model input data. Then, we combined the pose information to restore the workpiece point cloud. Finally, the prediction results were output with 3D bounding boxes and plotted them on 2D images through a series of trainings. However, as the main purpose of this study was to estimate the 6D pose of the workpiece, we used three of the evaluation metrics for the 6D pose estimation: one was the ADD metric for comparing the difference in the model points in the 3D space, the second was the 5 cm-5° metric that calculated the rotation and the translation error of the model in the 3D space, and the last was the Projection_2D metric that calculated the average distance difference of the 3D model points projected in the 2D plane. Finally, Projection_2D was used to calculate the average distance difference among the 3D model points in the 2D plane.

A. WORKPIECE 6D POSE ESTIMATION RESULTS

With the abovementioned three pose estimation indices to determine the accuracy of various workpiece pose predictions and to observe the curve of the loss function during the training process, we set the initial learning rate to 1 × $\mathbf{10^{-3}}$ and used the learning rate decay to gradually reduce the learning rate by half every 20 steps; the training frequency was set to 240 epoch.

1) CASE1: ADVANTECT UNO-2272G HIGH-PERFORMANCE AUTOMATION CALCULATOR

The first test piece was Advantech's UNO-2272G high-performance automatic computer, which captures RGB images with ArUco with the depth camera, and because ArUco has the feature of pose information, the current pose of each image was obtained through ArUco. In addition, the maximum diameter of the workpiece was measured to be 15.7 cm; this information was necessary to estimate the border of the workpiece within this range. In the training, Adam was used as the optimizer, the Momentum parameter was set to 0.9, and Batch Size was set to 4. In this study, the real pose of the workpiece at each view angle was obtained by pre-processing before training; the predicted pose of the workpiece by PVNet was expected to be closer to the real pose so that the accuracy of the system could be maintained at a certain level. The total loss function after 240-epoch training sessions is shown in Fig. 10(a), the classification loss function in Fig. 10(b), and the voting loss function in Fig. 10(c).

In addition, the 2D_Projection, ADD, and 5 cm-5° metrics were used to evaluate the accuracy of the predicted

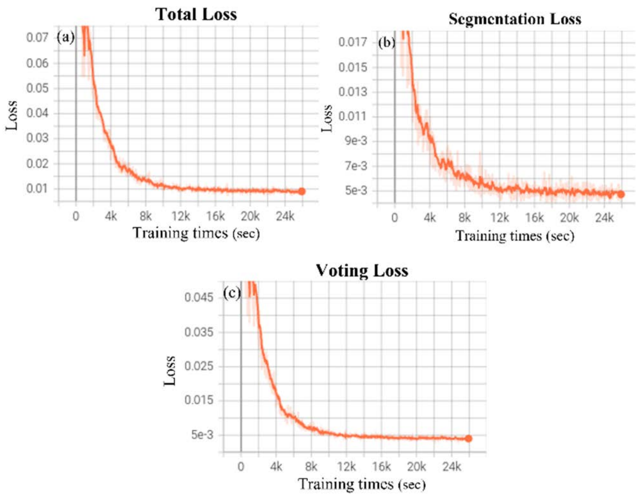

FIGURE 10. Case 1 loss function curve: (a) total loss function, (b) segmentation loss function, and (c) voting loss function.

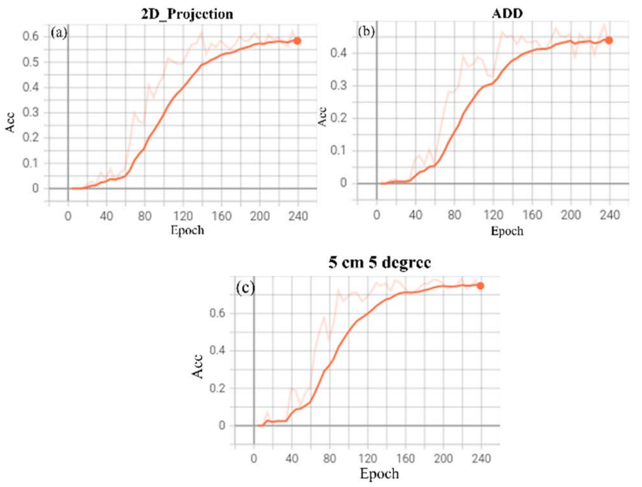

FIGURE 11. Case 1 evaluation indicator curve: (a) 2D_Projection, (b) ADD, and (c) 5 cm-5°.

attitude with respect to the real attitude under various conditions. As shown in Fig. 11(a), the accuracy of the 2D_Projection metric was 0.5781. Fig. 11(b) shows the accuracy of 0.4289 under the ADD indicator, and Fig. 11(c) shows the final accuracy of 0.7129 for the 5 cm-5° pointer, where the x-axis represents the number of operations and the y-axis represents the accuracy.

Moreover, the results of the 6D pose prediction bounding box for this workpiece are shown in Fig. 12, where the green bounding box denotes the ground truth and the blue bounding box represents the pose estimation.

2) CASE2: SHOE

In the shoe experiment, 500 RGB images with ArUco were acquired, and the maximum diameter of the workpiece was 26.5 cm, as determined by actual measurement. The total loss function after the 240-epoch training sessions is shown in Fig. 13(a), the categorical loss function in Fig. 13(b), and the voting loss function in Fig. 13(c), with the x-axis

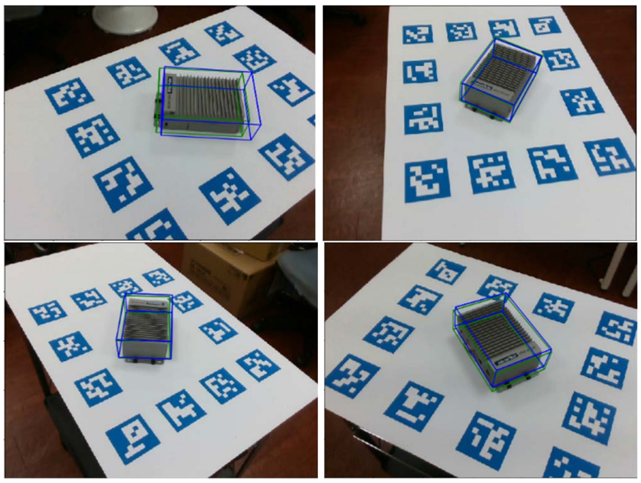

FIGURE 12. **Case 1: Ground-truth bounding box and pose estimation box results.**

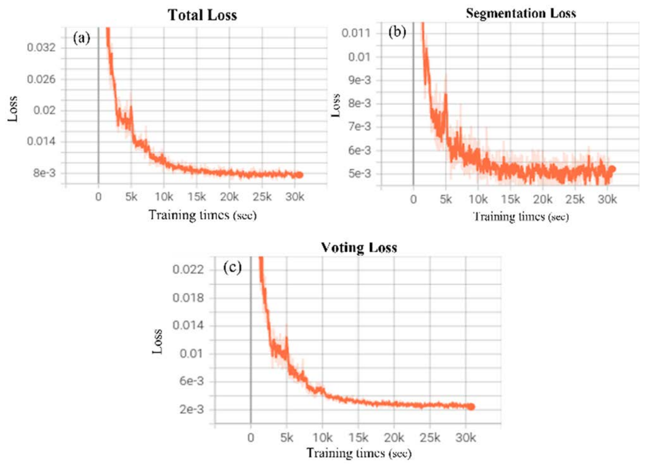

FIGURE 13. **Case 2 loss function curve: (a) total loss function, (b) segmentation loss function, and (c) voting loss function.**

coordinates representing the training time spent and the y-axis representing the loss function value.

From the graph, we can infer that the training curve in Case 2 oscillated at 3k–4k but then decreased until it converged at 10k–5k, and the total training time was approximately 30k, where the x-axis coordinates represent the training time spent and the y-axis represents the loss function value.

In the shoe experiment, the 2D_Projection, ADD, and 5 cm-5° metrics were evaluated. The curves are shown in Fig. 38, with the x-axis representing the number of training sessions and the y-axis representing the accuracy rate. Fig. 14(a) shows that the accuracy rate was 0.669 for 2D_Projection, while Fig. 14(b) shows that the accuracy rate was as high as 0.998 for ADD, and Fig. 14(c) shows that the accuracy rate was as high as 0.998 for the 5 cm-5° metric.

With the different metrics, we could demonstrate that the attitude estimation results in the 3D space were highly similar to the real conditions, thus proving that the completeness of the dataset enabled the system to obtain good estimation. We have presented the results of the bounding box for the 6D attitude prediction in Fig. 15.

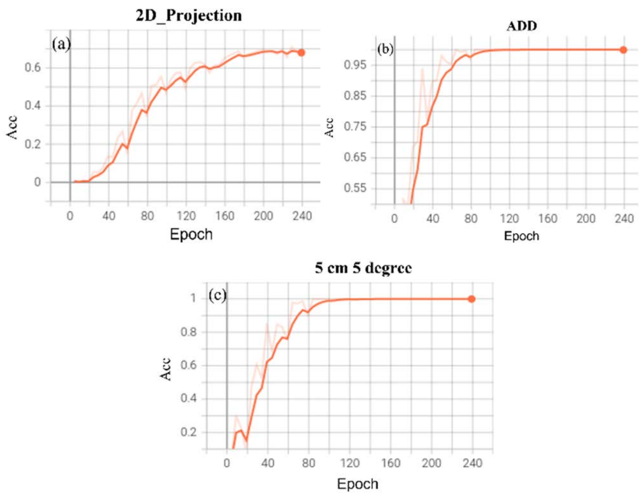

FIGURE 14. **Case 2 evaluation indicator curve: (a) 2D_Projection, (b) ADD, and (c) 5 cm-5°.**

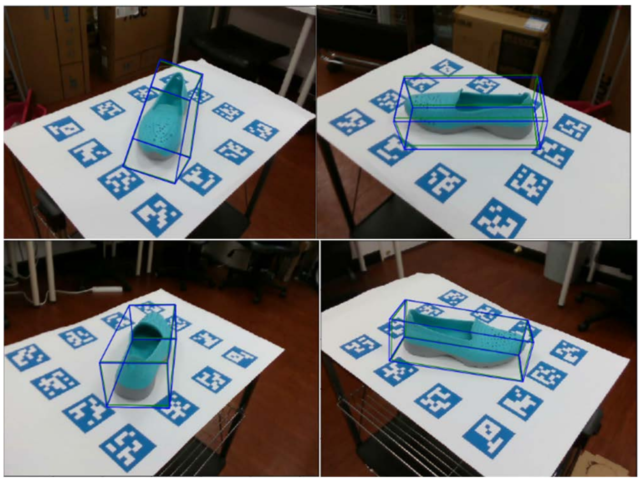

FIGURE 15. **Case 2: Ground-truth bounding box and pose estimation box results.**

3) CASE3: STAIRS

In all, 500 RGB images of ArUco were taken in the experiment for the ladder model, and the maximum diameter of the workpiece was measured to be 10 cm. the $1 \times \mathbf{10^{-3}}$, and the batch size was set to 4 for the training. The total loss function after the 240-epoch training sessions is shown in Fig. 16(a), the categorical loss function in Fig. 16(b), and the voting loss function in Fig. 16(c), where the x-axis coordinates represent the training time and the y-axis represents the loss function value.

In addition, in the experiment of the ladder model, the three metrics of 2D_Projection, ADD, and 5 cm-5° were evaluated, and the corresponding curves are shown in Fig. 17, with the x-axis representing the number of operations and the y-axis representing the accuracy. Fig. 17(a) shows the accuracy of 0.844 for the 2D_Projection metric after comparing the complete data, and Fig. 17(b) shows the accuracy of 0.976 for the ADD pointer. Finally, in Fig. 17(c), an accuracy of 1 was observed for the 5 cm-5° pointer.

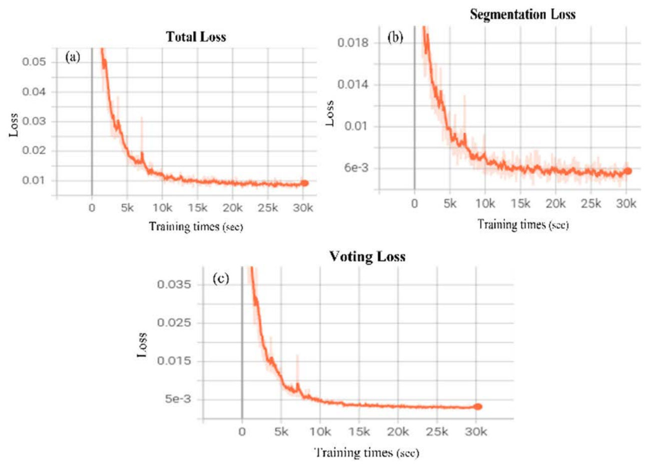

FIGURE 16. Case 3 loss function curve: (a) total loss function, (b) segmentation loss function, and (c) voting loss function.

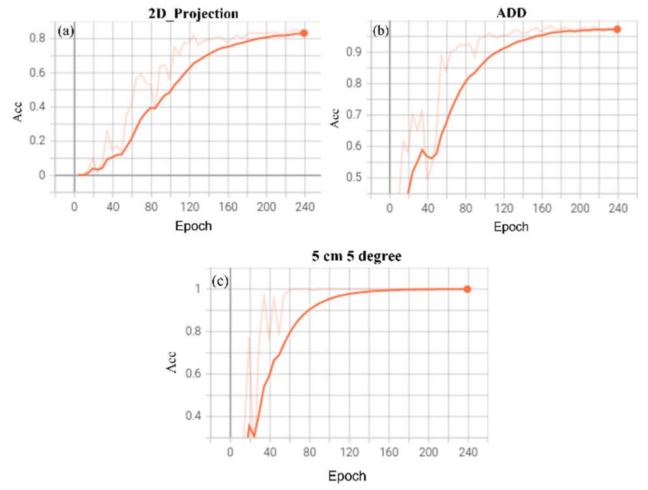

FIGURE 17. Case 3 evaluation indicator curve: (a) 2D_Projection, (b) ADD, and (c) 5 cm-5°.

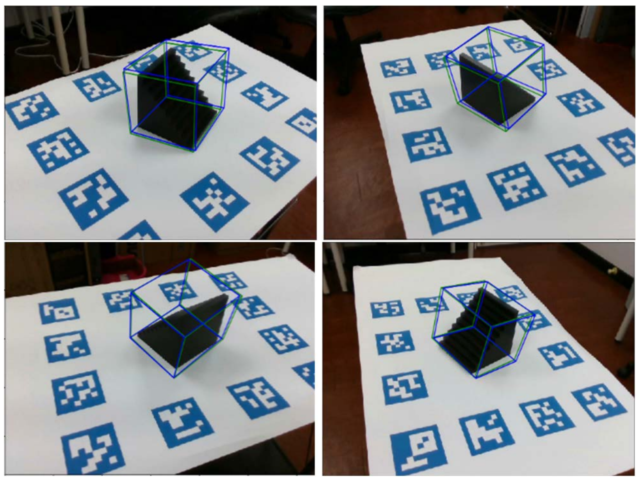

FIGURE 18. Case 3: Ground-truth bounding box and pose estimation box results.

TABLE 1. Results for model evaluation metrics.

Models	Evaluate Metrics		
	OUR		
	2D _Projection	ADD	5 cm-5°
Case1	0.578	0.428	0.712
Case2	0.669	0.998	0.998
Case3	0.844	0.976	0.99
Average	0.697	0.806	0.9

TABLE 2. Comparison evaluation metrics with different method.

Methods	Pose CNN[23]	Oberweger[24]	PVNet[7]	OURS
2D_Projection	37.2	39.4	47.4	**69.7**
ADD	61.0	72.8	73.4	**90**

The good performance of the three metrics in Case 3 with more than 85% accuracy could be attributed to the fact that the size of the model was more significant in terms of height than that in the previous cases; thus, the attitude in the 3D space or in 2D could be computed through the network model.

B. SYNTHESIS

In this study, the object observation PVNet network model could effectively perform the pose estimation task for all types of objects. The previous experiments by Peng et al. [7] demonstrated the robustness of this network model in the LINEMOD and OCCLUSION LINEMOD datasets. This study not only reproduced Peng et al.'s experiments to test the feasibility of the model but also focused on the creation of various types of images with different environmental conditions that could also be input into the PVNet model for training. However, the difference from Peng et al.'s experiment was that in this study, six objects were tested, the main training parameters were set to an initial learning rate of 0.001, and the learning rate was halved every 20 epochs until the total number of training sessions was 240 epochs. The accuracy of the experimental dataset was evaluated by three different pose estimation indices, as shown in Table 1. However, the average accuracy of the three evaluation indices in the overall experimental data was maintained at 70%.

In addition, to convert 2D flat images into the 3D space, solving the projection conversion matrix is an important computational step, an open dataset such as LINEMOD was chosen, we compared with other previous experimental results shown in Table 2. Our estimation results achieve better performance under the same pose evaluation metrics. The projection transformation matrix from the 2D plane and the 3D space could not be calculated effectively in this study. Therefore, in this study, we solved the problem of the projection transformation matrix, which was not explicitly mentioned in Peng et al.'s previous experiments [7], by inputting the corner points of the 3D model of the workpiece and the corner points of the workpiece in the 2D image plane and calculating the corresponding projection transformation matrix at different viewing angles with the PnP algorithm.

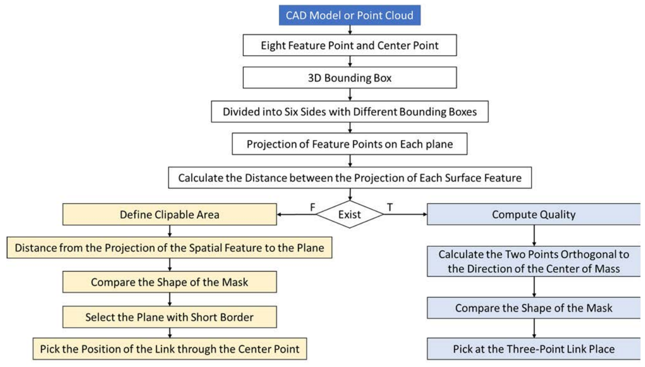
CAD Model or Point Cloud
Eight Feature Point and Center Point
3D Bounding Box
Divided into Six Sides with Different Bounding Boxes
Projection of Feature Points on Each plane
Calculate the Distance between the Projection of Each Surface Feature
Exist
F
T
Define Clipable Area
Distance from the Projection of the Spatial Feature to the Plane
Compare the Shape of the Mask
Select the Plane with Short Border
Pick the Position of the Link through the Center Point
Compute Quality
Calculate the Two Points Orthogonal to the Direction of the Center of Mass
Compare the Shape of the Mask
Pick at the Three-Point Link Place

FIGURE 19. Selection strategy flow chart.

C. APPLICATIONS

The ultimate goal was to allow the robot arm to achieve the clamping purpose in the correct way. Therefore, when there was a target workpiece posture, the next task was to select a suitable clamping position and direction, so that the robot arm could effectively clamp the target workpiece, according to the past reference data obtained using the following mainstream methods:

- Selection by a force analysis of the geometry of the workpiece model [25].
- Selection of the clamping area by means of human customary definitions [26].

It is possible to clamp the workpiece by using different strategic approaches. To alleviate the need to perform calibration before each clamping, we proposed a special clamping strategy from the 3D bounding box and feature information obtained from the 6D pose estimation results in this study, to guide the robot arm to adopt an automatic clamping strategy according to the pre-defined conditions before the action. The strategy flow chart is shown in Fig. 19.

In this strategy process, the 3D bounding box results and the feature point information from the previous pose estimation were used to find the clamping position according to the judgment conditions of each stage. In order for the robot arm to be able to clamp smoothly, we limited the clamping range to less than the retractable range of the fixture, so that all of the clamping tasks could be completed without changing the fixture. When there was a feature point distance that could be clamped in a certain plane, the calculation of the center of gravity was performed by the original point cloud of the workpiece. Therefore, we calculated two points that were orthogonal to the position of the center of mass. Then, we compared the masks in the direction of the view angle to determine the contour of the workpiece in this direction, and finally, we executed this strategy of clamping at a relatively smooth position with the center of mass.

In contrast, when there was no restriction on the distance of feature points in all the planes, another strategy was adopted to select the clamping position, and we defined the clamping area in the CAD model of the workpiece beforehand. In this strategy, the clamping point was selected as the best clamping point by comparing the pre-defined clamping area with the mask image, finding the relatively smooth part of the workpiece contour, calculating the angle with the center point, and selecting the clamping point with the smaller angle to the center point.

V. CONCLUSION

This study was based on PVNet for object pose estimation and feature extraction not only by using the previous public dataset on the Internet for testing but also for practical applications in any future factory production line. ArUco, in the real production line, sometimes does not have the CAD file of the workpiece itself in advance, but we could generate the workpiece point cloud file by registering, aligning, and segmenting the point cloud with the pose of each view angle image. However, most of the current processing methods are still simply using the traditional RGB image recognition technology, but it is not possible to obtain the required depth information, and only simple object segmentation and inspection can be performed, which makes it limited in many states and even causes the robot arm to be unable to accurately know the workpiece position and attitude during clamping tasks. Therefore, in this study, the 6D pose of the workpiece was finally estimated in real time by training the PVNet model, and the 3D boundary frame characteristic information applied to derive a clamping strategy are currently under study, and expected to be published soon.

REFERENCES

[1] L. Michalkova, V. Machova, and D. Carter, ''Digital twin-based product development and manufacturing processes in virtual space: Data visualization tools and techniques, cloud computing technologies, and cyber-physical production systems,'' *Econ., Manag. Financial Markets*, vol. 17, no. 2, pp. 37–51, 2022.

[2] M. Andronie, G. Lăzăroiu, M. Iatagan, C. Uţă, R. Ştefănescu, and M. Cocoşatu, ''Artificial intelligence-based decision-making algorithms, Internet of Things sensing networks, and deep learning-assisted smart process management in cyber-physical production systems,'' *Electronics*, vol. 10, no. 20, p. 2497, Oct. 2021.

[3] I. Ali, O. J. Suominen, E. R. Morales, and A. Gotchev, ''Multi-view camera pose estimation for robotic arm manipulation,'' *IEEE Access*, vol. 8, pp. 174305–174316, 2020.

[4] G. Lăzăroiu, M. Andronie, M. Iatagan, M. Geamănu, R. Ştefănescu, and I. Dijmărescu, ''Deep learning-assisted smart process planning, robotic wireless sensor networks, and geospatial big data management algorithms in the internet of manufacturing things,'' *ISPRS Int. J. Geo-Inf.*, vol. 11, no. 5, p. 277, Apr. 2022.

[5] W. Chamberlain, J. Leitner, T. Drummond, and P. Corke, ''A distributed robotic vision service,'' in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 2494–2499.

[6] L. Aing and W.-N. Lie, ''Detecting object surface keypoints from a single RGB image via deep learning network for 6-DoF pose estimation,'' *IEEE Access*, vol. 9, pp. 77729–77741, 2021.

[7] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, ''PVNet: Pixel-wise voting network for 6DoF pose estimation,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4561–4570.

[8] M. Metzner, S. Weissert, E. Karlidag, F. Albrecht, A. Blank, A. Mayr, and J. Franke, ''Virtual commissioning of 6 DoF pose estimation and robotic bin picking systems for industrial parts,'' *IFAC-PapersOnLine*, vol. 52, no. 10, pp. 160–164, 2019.

[9] C. Choi and H. I. Christensen, ''3D pose estimation of daily objects using an RGB-D camera,'' in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 3342–3349.

[10] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6D object pose prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 292–301.
[11] A. Kuzdeuov, M. Rubagotti, and H. A. Varol, "Neural network augmented sensor fusion for pose estimation of tensegrity manipulators," *IEEE Sensors J.*, vol. 20, no. 7, pp. 3655–3666, Apr. 2020.
[12] D. Jurado-Rodriguez, R. Munoz-Salinas, S. Garrido-Jurado, and R. Medina-Carnicer, "Design, detection, and tracking of customized fiducial markers," *IEEE Access*, vol. 9, pp. 140066–140078, 2021.
[13] *Pyrealsense2 Official Document File*. Accessed: Mar. 11, 2019. [Online]. Available: https://intelrealsense.github.io/librealsense/python_docs/_generated/pyrealsense2.html#modulepyrealsense2
[14] J. Cui, C. Min, X. Bai, and J. Cui, "An improved pose estimation method based on projection vector with noise error uncertainty," *IEEE Photon. J.*, vol. 11, no. 2, pp. 1–16, Apr. 2019.
[15] R. A. Boby and S. K. Saha, "Single image based camera calibration and pose estimation of the end-effector of a robot," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 2435–2440.
[16] J. Wu, "Rigid 3-D registration: A simple method free of SVD and eigendecomposition," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 10, pp. 8288–8303, Apr. 2020.
[17] F. Pomerleau, F. Colas, and R. Siegwart, "A review of point cloud registration algorithms for mobile robotics," *Found. Trends Robot.*, vol. 4, no. 1, pp. 1–104, May 2015.
[18] S. Arshad, M. Shahzad, Q. Riaz, and M. M. Fraz, "DPRNet: Deep 3D point based residual network for semantic segmentation and classification of 3D point clouds," *IEEE Access*, vol. 7, pp. 68892–68904, 2019.
[19] M.-O. Shin, G.-M. Oh, S.-W. Kim, and S.-W. Seo, "Real-time and accurate segmentation of 3-D point clouds based on Gaussian process regression," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 12, pp. 3363–3377, Dec. 2017.
[20] H. Chen, F. Xu, W. Liu, D. Sun, P. X. Liu, M. I. Menhas, and B. Ahmad, "3D reconstruction of unstructured objects using information from multiple sensors," *IEEE Sensors J.*, vol. 21, no. 23, pp. 26951–26963, Dec. 2021.
[21] D. Luhr, M. Adams, H. Houshiar, D. Borrmann, and A. Nuchter, "Feature detection with a constant FAR in sparse 3-D point cloud data," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 3, pp. 1877–1891, Mar. 2020.
[22] C.-Y. Tsai and S.-H. Tsai, "Simultaneous 3D object recognition and pose estimation based on RGB-D images," *IEEE Access*, vol. 6, pp. 28859–28869, 2018.
[23] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," in *Robotics: Science and Systems (RSS)*. 2018.
[24] M. Rad and V. Lepetit, "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3828–3896.
[25] R. Wang, J.-W. Chen, Y. Wang, L. Jiao, and M. Wang, "SAR image change detection via spatial metric learning with an improved Mahalanobis distance," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 1, pp. 77–81, Jan. 2020.
[26] H. Karimi-Rouzbahani, N. Bagheri, and R. Ebrahimpour, "Invariant object recognition is a personalized selection of invariant features in humans, not simply explained by hierarchical feed-forward vision models," *Sci. Rep.*, vol. 7, no. 1, pp. 1–24, Dec. 2017.

YI-HSIANG KAO was born in Taipei, Taiwan, in 1996. He received the M.S. degree in automatic control engineering from Feng Chia University, Taichung, Taiwan, in 2022. He is currently an AI Image Vision Development Engineer at AUO's Manufacturing Division. His research interests include 3D vision and robotic applications.

CHING-KUN CHEN was born in Changhua, Taiwan, in 1976. He received the M.S. degree from the Department of Automatic Control Engineering, Feng Chia University, Taichung, Taiwan, in 2000, and the Ph.D. degree from the Department of Electrical Engineering, National Chung-Hsing University, Taichung, in 2012. He is currently an Assistant Professor with the Department of Automatic Control Engineering, Feng-Chia University. His research interests include biomedical engineering, information security, and system integration.

CHIH-CHENG CHEN (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees from the Department of Mechatronics Engineering, National Changhua University of Education, in 2005 and 2011, respectively. He has been an Assistant Professor at Feng Chia University, Taiwan, since 2019. He has published 68 academic articles and owns three patents. He has also implemented several projects funded by the National Natural Science Foundation of China, Fujian Province. His research interests include AIoT technology and artificial intelligence, machine learning, and RFID applications.

CHEN-YEN LAN was born in Taipei, Taiwan, in 1976. He received the B.S. and M.S. degrees from the Department of Automatic Control Engineering, Feng Chia University, Taichung, Taiwan, in 1999 and 2001, respectively. He was a Software Engineer with the System Simulation Group, National Chung-Shan Institute of Science & Technology, Taichung, from 2001 to 2005. He is currently a Chief Executive Officer with professional automation system integration corporation, Node Health Technology Company Ltd., Taichung.