Low-Quality Rendering-Driven 6D Object Pose Estimation from Single RGB Image

Guoyu Zuo[*†], Chengwei Zhang[*†], Hongxing Liu[*†], Daoxiong Gong[*†]
[*]*Faculty of Information Technology*, *Beijing University of Technology*, Beijing, China
[†]*Beijing Key Laboratory of Computing Intelligence and Intelligent Systems*, Beijing, China
zuoguoyu@bjut.edu.cn, ZCW0356@emails.bjut.edu.cn, xingl@emails.bjut.edu.cn, gongdx@bjut.edu.cn

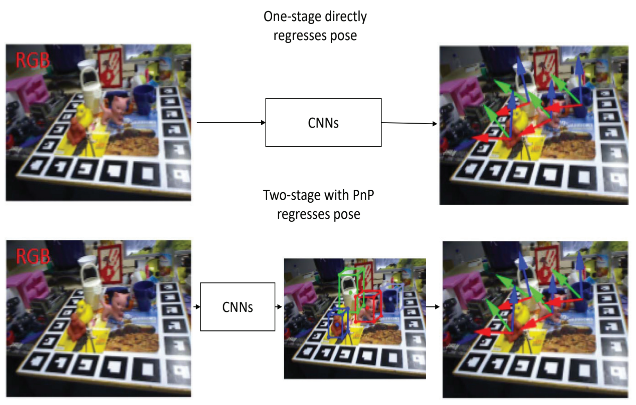
***Abstract*—The 6D object pose obtained from single RGB image has broad applications such as robotic manipulation and virtual reality. Among many existing methods, the deep learning-based approaches for object pose estimation from single RGB image are widely used. However, they often require a large amount of training data, which has great challenges in high cost of data collection and lack of 3D information. In this paper, we introduce an object pose estimation architecture that takes a single RGB image as input and directly outputs rotation angles and translation vectors. A data generation pipeline that applies the idea of domain randomization is used to generate millions of low-quality rendering images. Then the pose estimation is realized by fusing the architecture and the domain randomization approach to utilize the generated information and low the data collection cost. We synthesized a big dataset called Pose6DDR whose images are similar to those in the LineMod dataset. Experiments demonstrated the effectiveness of the proposed 6D object pose estimation architecture as compared to the relevant competing technologies.**

I. Introduction

Image-based 6D object pose estimation plays an increasingly important role in various applications, such as augmented reality, virtual reality, and robotic manipulation. For example, robust and accurate object pose estimation is crucial in Amazon Picking Challenge [1], where the robot needs to pick objects from a warehouse shelf. There are many challenges, including object detection under severe occlusion, variousness of lighting and appearances, and cluttered background objects. The traditional methods often establish the correspondences between the 3D models and their corresponding 2D images of objects. But they are more dependent on the hand-crafted features that are not robust to the cluttered backgrounds. The deep learning-based methods train neural networks to obtain the object pose estimation, which takes images as input and outputs its corresponding object pose, while their generalization ability is still a problem.

As is shown in Fig. 1 [7], the deep learning-based methods can be divided into two types: one-stage and two-stage. The two-stage approach uses the Perspective-n-Point(PnP) algorithm to recover the object 6D pose parameters [2], [3], [4], and the one-stage approach recovers the 6D pose without the processing of PnP. The two-stage approach detects the locations of 2D keypoints in image space since the 3D keypoints are known, and then computes the 6D object pose parameters with the PnP algorithm. The final effects depend on the detection accuracy of the 2D keypoints. This approach achieves the state-of-the-art performance due to the development of the detection methods of keypoints. However, these methods have difficulty in tackling occluded and truncated objects, for some keypoints may be unseen. Although Convolutional neural networks (CNNs) may predict these unseen keypoints by memorizing similar patterns, the generalization remains difficult. The one-stage approaches directly regress the 6D object pose. They locate the center in the image and predict the distance from the camera as their estimated 3D translation matrix, and regress the quaternion as their 3D rotation matrix, but they need post-refinement to improve their low accuracy.

This paper aims to recover the object pose from a single RGB image using the one-stage approach. Different from the aforementioned one-stage approach, the proposed method regresses the rotation angles and the translation vectors in


One-stage directly regresses pose
RGB
CNNs
Two-stage with PnP regresses pose
RGB
CNNs
Fig. 1. The flow charts of the deep learning-based methods. Top: one-stage method; Bottom: two-stage method.

This work is supported by the National Natural Science Foundation #61873008, and the Beijing Natural Science Foundation #4182008 and #4192010.

3D space directly. Compared with the above approaches, this method has a more simple network structure and easier training process. Although many approaches have been proposed to solve the problems such as cluttered scenes and occlusion, the challenge in the lack of training dataset is ignored. In other applications such as detection and autonomous driving, the simulated data are used to extend the training dataset. Hence, we extend this simulated data technology to 6D pose estimation, and improve the domain randomization approach to generate millions of training images for 6D pose estimation. These simulated data are expected able to improve the 6D pose estimation accuracy and further improve the robotic grasping.

There are some benchmark pose estimation datasets, such as LineMod [5], Occlusion LineMod [6], YCB-Video [16]. The proposed architecture was trained by using the self-built Pose6DDR dataset and LineMod datasets. Then it was evaluated on the Pose6DDR dataset and the wildly used LineMod dataset. The experiments demonstrated the effectiveness of the domain randomization-based dataset and the architecture to handle 6D object pose estimation.

This work has the following contributions:

- The domain randomization approach is used to generate the Pose6DDR dataset for 6D pose estimation with millions of images similar to the LineMod images.
- A 6D object pose estimation architecture is proposed to directly regress the rotation angles and the translation vectors of the object.
- The architecture trained on the generated dataset illustrates its generalization performance and robustness, and its test on the LineMod dataset also shows its better performance.

II. Related work

In this section, the 6D pose estimation methods and dataset extending approaches are reviewed briefly. The pose estimation methods mainly consist of template matching, keypoint correspondence and keypoint regression [7]. The dataset extending approaches include image enhancement, domain adaptation, and domain randomization.

A. 6D object pose estimation

1) Template-based methods: The template matching method can perform pose estimation on non-texture objects. First, the RGB-D images extracted from various directions of the obtained object model. Then these images are made into templates to match each position of the actual image. The matched pose is the estimated pose of the object. Hinterstoisser et al. [5] proposed a new image representation method by extending the image gradient direction for template matching and using a limited set of templates to represent 3D objects. By considering the 3D surface normal direction of the dense point cloud calculation obtained from the dense depth sensor, the accuracy of the estimated pose can be improved. Hodan et al. [8] proposed a method for detecting and accurately drawing 3D localization of multiple untextured and rigid objects in RGB-D images. By matching feature points in different modalities to verify the candidate object instances, each detected template associated with the approximate object poses are used as initial values for further optimization.

2) 2D-3D keypoints correspondence-based methods: When the model has rich textures, pose estimation can be performed by the method based on 2D-3D keypoints correspondence, and the model's 3D keypoints can be matched with its corresponding 2D keypoints. First, the existing 3D model is projected from various angles to render multiple images. Then, the correspondence between the 2D pixels and the 3D points is established by finding the matches between the 2D feature points in the observed image and the rendered image.

Lepetit et al. [9] used the PnP algorithm to obtain the 6D pose of the object by extracting the 2D feature points and the 3D feature points. And [2] presented a survey of 3D rigid object tracking methods based on monocular models. [11], [12], [13] used 3D descriptors to find the correspondence between the part of the 3D point cloud and the target's complete object 3D point cloud to obtain a rough pose estimate. Then it was refined by iterative closest point (ICP) [14] algorithm.

3) Regression-based methods: The regression-based 6D object pose estimation method directly recovers the 6D pose parameters of the target object from the input image. Usually, the method first detects the target in the image, and then combines it with the pose estimation [15]. The regression-based pose estimation has two ways. One is one-stage method, in which designed convolutional neural network takes one image as input for training, and solves the pose of the object by 3D rotation and 3D translation. The other is two-stage method, which is different from the one-stage method. This method first regresses the projection of the corresponding 3D keypoints of the target object in the 2D image and calculates the 6D object pose through the PnP algorithm. Rad et al. [26] predicted the 2D projection of their 3D bounding box corners and obtained a 2D-3D correspondence. Tekin et al. [10] proposed a deep CNN framework which used YOLOv2 to detect the 2D projection of the 8 bounding box corners in image space, and then used the PnP algorithm to calculate the 6D object pose. Yu et al. [16] proposed PoseCNN for the 6D object pose estimation. This method uses hough voting to determine the center of the object position and predict the distance from the camera to estimate the 3D translation of the target. The 3D rotation is calculated by returning the quaternion. PoseCNN proposes a new loss function, ShapeMatch-Loss, which can be applied to pose estimation of rotationally symmetric objects. And in their work, they proposed a new dataset: YCB-Video dataset. Do et al. [17] proposed an end-to-end deep learning framework named Deep-6DPose, which jointly detects, segments, and recovers 6D poses of object instances from a single RGB image. Liu et al. [18] proposed a two-stage CNN architecture that directly outputs the 6D pose without requiring multiple stages and additional post-processing.

Hu et al. [19] proposed a segmentation-driven 6D pose estimation framework in which each visible part of the object contributes to the local pose prediction in the form of 2D keypoints positions. The pose candidates are combined into a set of reliable 2D-to-3D correspondences, and then the 6D object pose estimation is calculated by the PnP algorithm. Peng et al. [20] proposed a 6D object pose estimation based on a pixel-level voting network architecture, in which the local information of the visible part of the object is used to detect keypoints for the occluded object. This method can detect the 6D pose of the target object in the truncated state. It has become the state-of-the-art method in the pose estimation area.

B. Extending the dataset

For the problem of lacking training data and high data acquisition cost, the image data for training can be collected by simulation, image enhancement, and etc. It is often faster to fine-tune a controller learned in simulation than to learn from scratch in the real world [23], [24]. However, the factors such as lighting conditions and textures in the real environment are difficult to fully reproduce in the simulation environment. There is a large gap between simulation and real images, and it is not ideal to train the model by using simulated images instead of real images. The method to reduce the difference between simulation and real image data is to use a high-fidelity simulation environment to render pictures. For example, Kanade et al. [21] uses high-simulation synthetic images for target perspective evaluation, but this method has poor performance in complex scenes. The high computing resources and the high quality of the model are required. The domain randomization approach is such a method to reduce the difference between simulation and real image data. Some work has previously explored the idea of using domain randomization to bridge the reality gap. Tobin et al. [22] uses the domain randomization approach to generate the model and images to train the autoregressive model to grasp the target and demonstrate the effectiveness of the domain randomization.

III. Method

The goal of our paper is to learn a architecture that using single RGB images synthesized by the proposed domain randomization approach and outputs the 6D object pose estimation for robotic grasping. The current 6D pose estimation methods face the problem of small datasets. To improve the accuracy of pose estimation, we generated a dataset with millions of images which are similar to the LineMod images. Compared with the architecture trained only with the LineMod images, the network structure trained with the generated images has better generalization ability, because these images are generated with more scenes.

A. The Pose6DDR dataset

The self-built dataset called Pose6DDR dataset is introduced in the following parts, including the generation of objects, the label annotation and the randomization factors.

1) object generation: First, eight types of models were collected for generating the images by Bullet physics engine: ape, can, cat, driller, duck, eggbox, glue and holepuncher. The scale of the object is the same as the model in the common dataset. Then, the domain randomization approach is used to bridge the gap between the synthesized and the real images. Next, the distribution of the model was generated by the low-quality rendering in the synthesized images, in order to match the distribution of the model in the LineMod dataset. So the synthesized images are more like the real images, as can be seen in Fig. 2.

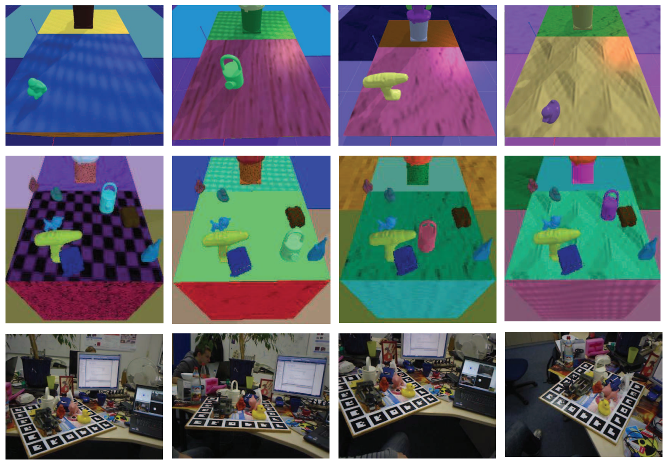


Fig. 2. Image examples. The first row is the object models, the middle row is the Pose6DDR dataset images and the last row is the common dataset images.

2) Annotation: The Pose6DDR dataset is used as training set to train the architecture. Labels of the images are annotated in the process of generating the dataset. The architecture obtains the class, the rotation angles of roll, pitch and yaw, and the translation vectors of the object. The Pose6DDR dataset is annotated with 16 numbers, which contain class (1), 3 angles (3*1), rotation matrix (9), and translation matrix (3). The classification is represented by one-hot encoding. The rotation angles and translation vectors are computed as in Fig. 3.

In order to set the label information for the data, the 8 corners coordinates of the 3D bounding box need to be derived. As in Fig. 3, the square $ABCD$ is the top view of the 3D bounding box before rotation, and the square $A^{'}B^{'}C^{'}D^{'}$ is the top view of the bounding box rotated by a certain angle θ and θ=$\angle B^{'}EB$, r is the length of the bottom side of the 3D

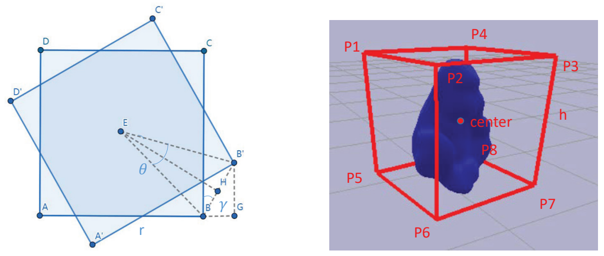


Fig. 3. The process diagram of label information annotation.

bounding box, and h is the height of the 3D bounding box, then:

$$EB = \frac{\sqrt{2}}{2} r \tag{1}$$

$$BH = EB * \sin\frac{\theta}{2} \tag{2}$$

$$BB^{'} = 2BH = 2\frac{\sqrt{2}}{2} r * \sin\frac{\theta}{2} \tag{3}$$

$$\angle EBB^{'} = \angle BB^{'}G = \frac{180-\theta}{2} - 45 \tag{4}$$

Let l = $BB^{'}$, $\gamma = \angle CBB^{'} = \angle BB^{'}G$, and the coordinates of point B be (x_B, y_B), then the coordinate of point $B^{'}$ is $(x_B - \cos\gamma * l, y_B + \sin\gamma * l)$. The center point of the target object coincides with the center of the 3D bounding box. The position of each point in space can be seen in Fig. 3. Let the coordinates of the center point of the target object be (x, y, z). The coordinates of the 8 corners of the 3D bounding box can be obtained by similar transformations, respectively.

3) randomization factors: The main idea is to generate Pose6DDR dataset by changing the factors in the image space with low-quality rendering. Specifically, the approach randomly generates the texture of workbench and manipulator, position, illumination, direction, and so on.

- Content Variation: More than 8 objects are placed in a 3D background. To achieve the images similar to the LineMod dataset, more than eight different types of object models were used consistent with the model scale of the LineMod dataset. We generated the data of one type of object with random directions and positions by fixing other type of objects in the generated images to make them more similar to the images in the LineMod dataset.
- Style Variation: Style variation is obtained by randomly changing the colors and textures of all the objects. The texture library with 8 different textures is used to realize style variation. In addition, light enhancement is applied to capture varying shadow conditions and temporal variations. The changes of illumination conditions are realized by changing the position and orientation of the light source.

In Fig. 2, the middle row is our generated images, and the last row is the LineMod images. We can see that the synthesized data is similar to the LineMod data. The LineMod has more than 1213*8 images, while our dataset has millions of images which were generated by using our method in two whole days. The following experiments validated the effect of the generated dataset. To some extent, the proposed Pose6DDR dataset can improve the accuracy of pose estimation and reduce image collection costs.

B. 6D Object Pose Estimation Architecture

1) Pose Representation: There are many challenges to regress the 6D object pose directly. For example, the category-based 6D object pose estimation has the limitations such as lack of 3D model information and need for multi-stage processing. However, the current RGB datasets have not enough RGB images with fully 3D space representation used as the input of many architectures. To achieve the pose estimation goal, we first need to represent the pose information. In our method, the translation matrix is represented by the outputs of the architecture. The rotation matrix is more complicated than the translation matrix but it can be resolved by the rotation angles. For the rotation angles wrap around at 2π radians, the same angle can be represented by multiple values. To void the problem, we take the minimum of the obtained values as the rotation angle. Then the angles and vectors are used to compute the rotation matrix and the translation matrix. The rotation matrix of the object is calculated by Eqs. (5)-(7).

$$R_x(\varphi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi \\ 0 & -\sin\varphi & \cos\varphi \end{bmatrix} \tag{5}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{6}$$

$$R_z(\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

where φ, θ, ψ are the roll, pitch and yaw angles of the object. Then the 3D rotation matrix is represent by R = $R_x \cdot R_y \cdot R_z$.

2) Network Architecture: Fig. 4 depicts the main architecture, which has two parts: the backbone and the pose estimation parts. The backbone is used to extract features of input images and shared to the next part. The pose estimation part contains two streams, one is the classification stream, and the other is a pose estimation stream. The classification

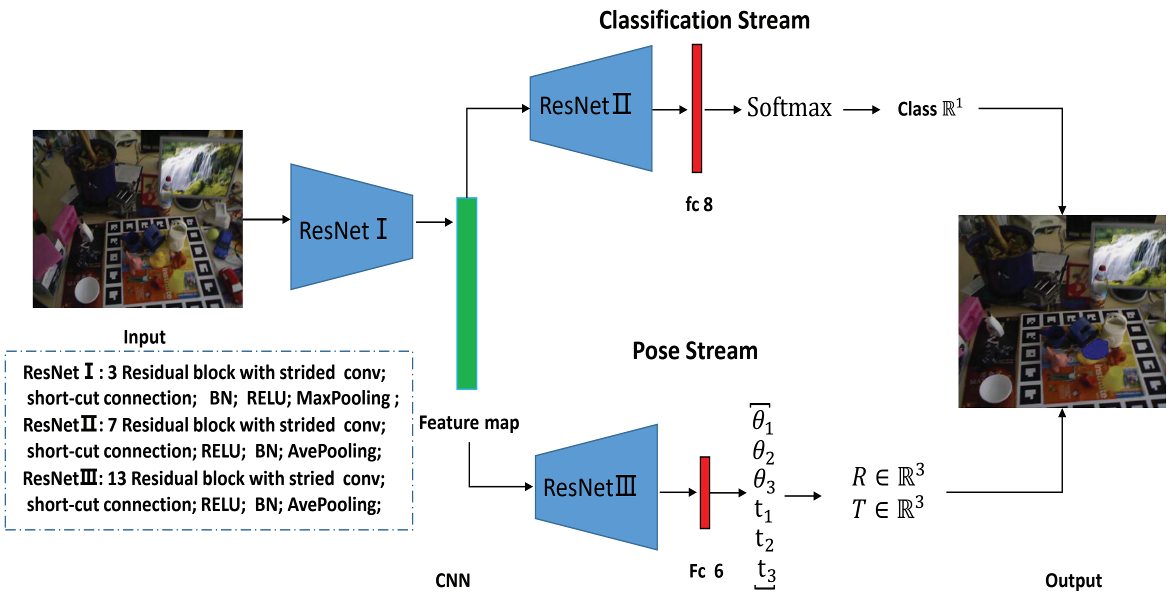

Classification Stream
ResNet Ⅱ
Softmax
Class $\mathbb{R}^1$
fc 8
ResNet Ⅰ
Input
ResNet Ⅰ : 3 Residual block with strided conv; short-cut connection; BN; RELU; MaxPooling;
ResNet Ⅱ: 7 Residual block with strided conv; short-cut connection; RELU; BN; AvePooling;
ResNet Ⅲ: 13 Residual block with stried conv; short-cut connection; RELU; BN; AvePooling;
Feature map
Pose Stream
ResNet Ⅲ
Fc 6
θ_1 θ_2 θ_3 t_1 t_2 t_3
$R \in \mathbb{R}^3$
$T \in \mathbb{R}^3$
CNN
Output

Fig. 4. The architecture of our method: The first row is the class stream, the second row is the pose estimation stream, and they are connected to a shared part based on ResNet. The part takes a single RGB image as input. The first shared ResNetI model including three residual blocks with two $3 * 3$ convolutions, stride $= 2$ extracts the feature map. For the class stream, ResNetII has seven residual blocks with one $1 * 1$ convolution, stride $= 1$ and a fully connected layer. For the pose stream, ResNetIII have thirteen residual blocks and every block contains two $1 * 1$ convolutions, stride $= 1$. And at the end of the pose stream followed by a fully connected layer for the six parameters of three angles and three translation numbers.

stream predicts the class of the object. The pose estimation stream predicts the angles and the translation vectors of the object in 3D space to compute their rotation and translation matrices. The ResNet was chosen as the main architecture for its better effect after comparing with different networks.

3) multi-object loss function: A multi-task loss is defined to train the classification, the rotation angles, and the translation vectors. The loss function is represented as follows:

$$\ell = \alpha_1 \ell_{cls} + \alpha_2 \ell_{pose} \tag{8}$$

where the classification loss ℓ_{cls} is defined as the $softmax$ loss and ℓ_{pose} is the pose loss; α_1, α_2 are the scale factors to control the importance of each loss in the training process. The pose estimation stream outputs 6 vectors, which present the angles and the translations. The pose regression loss ℓ_{pose} is defined as follows:

$$\ell_{pose} = \|r - \hat{r}\| + \beta \|t - \hat{t}\| \tag{9}$$

where r are the regressed rotation angles and $\hat{r}$ are the ground truth of the angles; t are the output of translation vector, and $\hat{t}$ are the ground truth of the translation, and β is the scale factor to control the positions and the translation regression errors.

C. Training and test

During the training and testing processes. We implemented the PyTorch-based network architecture with single RGB image as input. The outputs of the architecture include the classification and the 6 vectors of the object, which are used to compute the rotation matrix and the translation matrix. This design allows the architecture to make full use of 3D spatial information of images. To be mentioned, α_1, α_2 and β in Eqs. (8) and (9) are empirically set to 0.7, 1 and 0.7, respectively. According to the different effects of the two streams in the network, the role of ℓ_{pose} improved by setting ℓ_{pose} as 1 to obtain better performance.

The architecture is trained on a GTX 1080Ti GPU for more than 200 epochs by using an Adam optimizer with a momentum of 0.1 and a weight decay of 0.0001. Each mini-batch has 128 images, and the learning rate of the first 20 epochs sets to 0.0001.

IV. Experiments

To evaluate the effectiveness and robustness of our proposed architecture and Pose6DDR dataset, we validate the architecture with Pose6DDR dataset and LineMod dataset. The public dataset LineMod is the single object pose dataset. In LineMod dataset, every RGB image has a ground truth rotation matrix, translation matrix, and ID. There are 13 objects and 3D models for instance-level 6D object pose estimation. And each object has almost 1214 images. Different from LineMod dataset, the Pose6DDR dataset has more than 80000 images.

we use the $5cm5^\circ$ and ADD metrics to evaluate the performance of our method by calculating the average distance between the predicted pose and those obtained with the ground truth. In $5cm5^\circ$ metric, the estimated pose is accepted if it falls within $5cm$ translation and 5° rotation error compared with the ground truth. In ADD metric, the estimated pose is accepted if the average distance between the estimated pose and the transformed model point clouds by the ground truth pose is smaller than 10% of the object diameter. The ADD metric is computed as follows:

$$s = \frac{1}{|M|} \sum_{x_1 \in M} \min_{M} \|(Rx + t) - (\hat{R}x + \hat{t})\| \tag{10}$$

where R and t are the ground truth rotation and translation matrices, and $\hat{R}$ and $\hat{t}$ are the predicted rotation and translation matrices. M is the vertex of the 3D model.

A. Results on the Pose6DDR dataset

We tested the effectiveness of the architecture on the Pose6DDR dataset, and only the RGB images used for training and test, in which 2000 synthesized images are used for test. Besides $5cm5^\circ$, the other metric scales are used to test the robustness of the architecture by reducing the angle and the distance. Table I shows the results on Pose6DDR dataset with different metric scales.

TABLE I
RESULTS OF THE SIMULATED IMAGES WITH DIFFERENT SCALES.

target object	$5cm5^\circ$	$3cm5^\circ$	$1cm5^\circ$	$5cm3^\circ$	$5cm1^\circ$
ape	**96.9**	89.1	64.0	4.1	0
can	**98.3**	77.6	33.6	**98.3**	2.9
cat	**98.5**	93.7	83.9	82.1	0
driller	**99.6**	97.4	94.5	0	0
duck	**93.3**	76.4	47.5	48.2	1.2
eggbox	**74.8**	52.2	25.2	60.0	1.3
glue	**96.2**	90.2	65.9	15.0	0.2
holepuncher	**96.1**	78.8	62.8	**96.1**	0
average	**94.2**	81.9	59.7	50.5	0.7

The first column is the results of the $5cm5^\circ$ metric. It can be seen that the accuracy is almost over 95% with the exception of 74.8% for eggbox and 93.3% for duck. The last four columns are the results with the stricter metrics of $5cm3^\circ$, $5cm1^\circ$, $3cm5^\circ$, and $1cm5^\circ$, respectively. Because the generated dataset has a large amount of data and high similarity to traditional data set images, the network architecture trained on the dataset can achieve high accuracy rate mentioned in the table. It can be seen that the angle of the object has a greater impact on the accuracy results when the metric changes. In other words, the robustness of the architecture is poor in angle. However the distance metric is more robust compared with the direction. It is noted that the driller has results of two zeros, for the predicted rotation angles are between 3° and 5°. Fig. 5 illustrates the pose estimation results on the synthesized images.

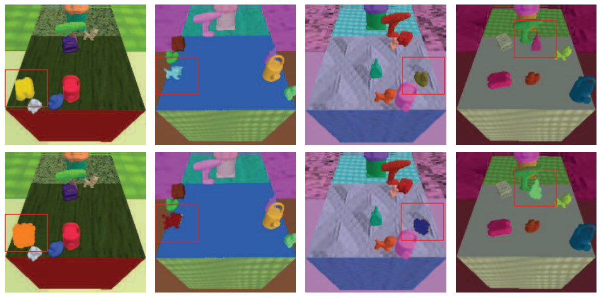


Fig. 5. Results on four synthesized images. The first row: synthesized images. The second row: the pose results. The red box is used to mark the target object.

B. Results on LineMod dataset

To test the effectiveness of our architecture, it is evaluated with the LineMod dataset and compares it with some other methods with the $5cm5^\circ$ and ADD metric. In experiments, there are 200 images of one kind of object used for test. For our method do not have the postprocessing, we compared it with the current methods without postprocessing including Brachmann [25], BB8 [26] and Deep6DPose [17]. These methods were evaluated with the $5cm5^\circ$ and ADD metrics on the LineMod dataset without refinement. Table II and Table III show the pose estimation results with the two metrics.

From the table II: the $5cm5^\circ$ metric, it can be seen that the average accuracy with these 8 objects of BB8 is 3.3% higher than Deep6DPose, while the Deep6DPose is 27.5% higher than Brachmann. The Deep6DPose is more stable than BB8 and Brachmann. It is noted that the results of our method with $5cm5^\circ$ is slightly lower than Deep6DPose on the average and 12.3% higher than Brachmann. Moreover, in the results of our method, the accuracy of some objects is higher than Deep6DPose and BB8, such as eggbox with 92.0%. Our results outperform the BB8 with the duck and eggbox. Relatively, our method has the worst result for the glue. By experiment, we found that the predicted translation of glue has a large gap with the ground truth. Compared with the Deep6DPose, we can also find that the results of our method are not stable enough. Although on the average, the proposed method is not higher than all mentioned approaches, it has outperformed some approaches and achieved highest accuracy for ape, duck and eggbox. Particularly, the architecture trained with the LineMod dataset spent more time than Pose6DDR dataset, and the method trained with the BJIT-dataset has a higher performance. We will further explore the expressiveness of generating data and strive to achieve the ability to replace the LineMod dataset.

Table III shows the results of the 8 objects with the ADD metric. We can see that the average accuracy of Deep6DPose outperforms BB8 by 22.9%. Our method has also outperformed with BB8 and Brachmann on the average and slightly

TABLE II
THE POSE ESTIMATION RESULTS WITH THE $5cm5^\circ$ METRIC.

target object	ape	can	cat	driller	duck	eggbox	glue	holepuncher	average
Deep6DPose [17]	57.8	70.1	70.3	72.9	67.1	68.4	64.6	70.4	67.7
Brachmann [25]	34.4	48.4	34.6	54.5	22.0	57.1	23.6	47.3	40.2
BB8 [26]w/Ref.	80.2	76.8	79.9	69.6	53.2	81.3	54.0	73.1	71.0
ours	57.5	63.0	45.0	54.0	56.5	**92.5**	38.0	47.5	56.8

TABLE III
THE POSE ESTIMATION RESULTS WITH THE ADD METRIC.

target object	ape	can	cat	driller	duck	eggbox	glue	holepuncher	average
Deep6DPose [17]	38.8	86.1	66.2	82.3	32.5	79.4	63.7	56.4	63.2
Brachmann [25]	33.2	62.9	42.7	61.9	30.2	49.9	31.2	52.8	45.6
BB8 [26]	27.9	48.1	45.2	58.6	32.8	40.0	27.0	42.4	40.3
ours	**49.5**	73.0	51.0	42.0	**49.0**	**79.5**	52.5	47.0	**55.4**

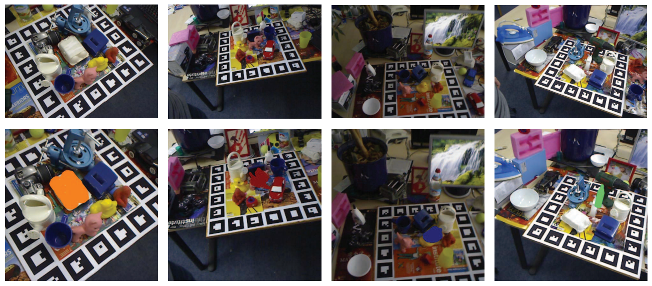


Fig. 6. Visualization of the single object pose estimation. The first row: LineMod images. The second row: the pose results of eggbox, cat, duck and glue.

lower than Deep6DPose. The duck in all methods has low accuracy, but our result with 49.0% is higher than others. On the whole, our proposed method has competitive accuracy. Furthermore, the results under the ADD metric are more stable than others. Fig. 6 shows the results of single object pose estimation on the LineMod dataset.

V. CONCLUSION

In this paper, we propose a 6D object pose estimation architecture for recovering 6D pose of the object from a single RGB image. The 6D architecture can directly output estimated rotation angles and translation vectors and calculate the 6D poses. The novelty of the architecture is to use the rotation angles to represent the rotation space. Besides, the low-quality rendering technique is use to design the Pose6DDR dataset with more than millions images in low collection cost. The 6D architecture can be trained by the simulated dataset and has an excellent performance to valid the effectiveness of the dataset. Experiments compared the proposed method with the state-of-the-art RGB-based 6D object pose estimation methods, and results show the effectiveness of our method. In the next work, we will extend the dataset with different approaches and improve the performance of the network in handling images in more complicated scenes.

REFERENCES

[1] Correll N , Bekris K E , Berenson D , et al. “Analysis and Observations from the First Amazon Picking Challenge”. IEEE Transactions on Automation Science & Engineering, pp. 15(1):172-188, 2016.
[2] Vincent Lepetit and Pascal Fua. “Monocular Model-Based 3D Tracking of Rigid Objects: A Survey”. Foundations and Trends® in Computer Graphics and Vision, pp. 1(1):1-89, 2005.
[3] Fred Rothganger, Svetlana Lazebnik, et al. “3D Object Modeling and Recognition Using Local Affine-Invariant Image Descriptors and Multi-View Spatial Constraints”. International Journal of Computer Vision, pp. 66(3):231-259, 2006.
[4] Wagner D , Reitmayr G , Mulloni A , et al. “Pose tracking from natural features on mobile phones”. In: Proc. Ismar, pp. 125-134, 2008.
[5] Stefan Hinterstoisser, Vincent Lepetit, et al. “Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes”. In Asian conference on computer vision, pp. 548-562, Springer, 2012.
[6] Eric Brachmann, Alexander Krull, et al. “Learning 6D object pose estimation using 3D object coordinates”. In European conference on computer vision, pp. 536-551, Springer, 2014.
[7] Guoguang Du, Kai Wang, et al. “Vision-based Robotic Grasping from Object Localization, Pose Estimation, Grasp Detection to Motion Planning: A Review”. arXiv preprint arXiv:1905.06658, 2019.
[8] Tomáš Hodaň, Xenophon Zabulis, “Detection and fine 3D pose estimation of texture-less objects in RGB-D images”. In: Proc. IROS, pp. 4421-4428, 2015.
[9] Vincent Lepetit, Francesc Moreno-Noguer, et al. “Epnp: An accurate o(n) solution to the pnp problem”. In: Proc. IJCV, pp. 81(2):155-166, February 2009.
[10] Bugra Tekin, Sudipta N. Sinha, et al. “Real-Time Seamless Single Shot 6D Object Pose Prediction” In: Proc. CVPR, pp.292-301, 2018.
[11] R. B. Rusu, N. Blodow, and M. Beetz. “Fast point feature histograms (fpfh) for 3d registration”. In: Proc. ICRA, pp. 3212-3217, May 2009.
[12] Andrew E Johnson. “Spin-images: a representation for 3-d surface matching”, 1997.
[13] Samuele Salti, Federico Tombari, and Luigi Di Stefano. “Shot: Unique signatures of histograms for surface and texture description”. Computer Vision and Image Understanding, pp. 125:251-264, 2014.
[14] Paul J. Besl and Neil D. McKay. “A method for registration of 3-d shapes”. IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 14(2):239-256, February 1992.

[15] Aniruddha V Patil and Pankaj Rabha. “A survey on joint object detection and pose estimation using monocular vision”. arXiv preprint arXiv:1811.10216, 2018.

[16] Yu Xiang, Tanner Schmidt, et al. “Posecnn: A convolutional neural network for 6D object pose estimation in cluttered scenes”. arXiv preprint arXiv:1711.00199, 2017.

[17] Thanh-Toan Do, Ming Cai, et al. “Deep-6dpose: recovering 6d object pose from a single rgb image”. arXiv preprint arXiv:1802.10367, 2018.

[18] Fuchang Liu, Pengfei Fang, et al. “Recovering 6D object pose from rgb indoor image based on two-stage detection network with multi-task loss”. Neurocomputing, pp.337:15-23, 2019.

[19] Hu Y, Hugonot J, et al. “Segmentation-driven 6D Object Pose Estimation”. In: Proc. CVPR, pp.19-24, 2019.

[20] Peng S, Liu Y, Huang Q, et al. “PVNet: Pixel-wise Voting Network for 6DoF Pose Estimation”. In: Proc. CVPR, 2019.

[21] Movshovitz-Attias Y, Kanade T, et al. “How useful is photo-realistic rendering for visual learning?”. In: Proc. ECCV, pp. 202-217, 2016.

[22] Tobin J, Biewald L, et al. “Domain Randomization and Generative Models for Robotic Grasping”. In Proc. IROS, pp. 3482-3489, 2018.

[23] Mark Cutler and Jonathan P How, et al. “Efficient reinforcement learning for robots using informative simulated priors”. In: Proc. ICRA, pp. 2605-2612, 2015.

[24] J Zico Kolter and Andrew Y Ng, et al. “Learning omnidirectional path following using dimensionality reduction”. In Robotics: Science and Systems, pp. 27-30, 2007.

[25] Eric Brachmann, Frank Michel, et al. “Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image”. In: Proc. CVPR, pp. 3364-3372, 2016.

[26] Mahdi Rad and Vincent Lepetit, et al. “BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth”. In: Proc. ICCV, pp. 3848-3856, 2017.