E2EK: End-to-End Regression Network Based on Keypoint for 6D Pose Estimation

Shifeng Lin[1*], Zunran Wang[2*], Yonggen Ling[2], Yidan Tao[3], Chenguang Yang[4]

***Abstract*—The methods based on deep learning are the mainstream of 6D object pose estimation, which mainly include direct regression and two-stage pipelines. The former are keen by many scholars at first due to their simplicity and differentiability to poses, but they usually lack in accuracy when compared with the latter that estimate the intermediate variables relating to geometries such as object keypoints or 2D-3D correspondence before PnP/RANSAC algorithm. However, the loss function of the two-stage method is non-differentiable to the 6D pose, which is hard to apply in the tasks requiring the differentiable poses. To overcome the disadvantages of the above methods, we propose an end-to-end regression network based on keypoints for 6D pose estimation. Specifically, we supervise the point-wise keypoint offsets that help the network to learn the geometric information and directly regress the 6D pose through aggregating keypoints to achieve differentiability to the pose. Furthermore, we improve the sampling method by sampling points around objects that benefits the small object and design a unit loss function that helps the learning of the keypoints. Experimental results show that our approach outperforms most methods on LM, LM-O and YCB-V datasets.**

***Index Terms*—Deep Learning for Visual Perception, RGB-D Perception, Pose Estimation**

I. INTRODUCTION

6D object pose estimation can be employed in lots of real-world applications such as robot grasping [42], [45], augmented reality [43], automatic driving [44] and information transfer [34]. In particular, the high-precision result of pose estimation conducts to a more stable robot grasping [42], [45]. Recently, this task has gained substantial achievement with the advent of deep learning [4]–[6], [8]. Early approaches regard pose estimation as a discrete space pose classification problem, but they only get rough results [11], [18]. Other methods construct completely differentiable networks to directly regress the 6D pose [7], [15], [16](Figure 1(a)). Although they have fast speed and end-to-end trainability, they get poor performance on accuracy because of only single pose supervision information [3], [7].

Manuscript received Dec 17, 2021; Revised March 16, 2022; Accepted April 20, 2022.

This paper was recommended for publication by Editor Cesar Cadena Lerma upon evaluation of the Associate Editor and Reviewers' comments.

[1] is with the school of Automation Science and Engineering, South China University of Technology, Guangdong, China sf_lin@qq.com. [2] are with Robotics X, Tencent zran.wang@qq.com; rolandling@tencent.com. [3] is with Shanghai Jiao Tong University and City University of Hong Kong yidantao@outlook.com. [4] is the corresponding author and with Bristol Robotics Laboratory, University of the West of England, Bristol, BS16 1QY, UK cyang@ieee.org.

∗ are equal contribution. This work is done when Shifeng Lin is an intern in Tencent Robotics X.

Digital Object Identifier (DOI): see top of this page.

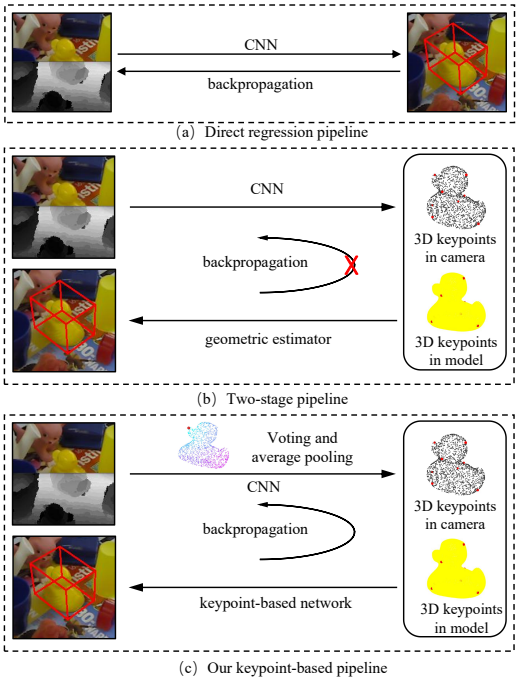


Fig. 1. Overview of the proposed method. (a) shows the direct regression method which has a fast speed and end-to-end trainability but suffers poor performance. (b) shows the two-stage pipeline which has a better performance but is non-differentiable to the 6D pose. (c) represents our regression network based on keypoints. Specifically, we remove the outlier through the correlation between the predicted keypoints and integrate an end-to-end 6D pose estimator into keypoint regression, which achieves differentiability to the pose while taking into account speed and accuracy.

Mainstream researchers try to tackle this problem by dividing the task into two stages. They first introduce the intermediate variables to mine the potential information of the pose and then solve the pose through the geometric estimator(Figure 1(b)). For example, [4], [6] predict the dense 2D-3D correspondence and then solve the pose by the PnP/RANSAC. [2] predicts the point-wise offset from the visual part to the keypoints. The keypoints are then obtained by voting and the pose is solved by PnP. Although the two-stage methods provide better estimation results, there are still some problems. Firstly, these methods usually use the corresponding relationship of intermediate variables as the objective function for training. However, the intermediate variables cannot directly reflect the error of the pose estimation, for the error of the inferred pose from two different sets of intermediate variables may be the same. Secondly, these methods are not differentiable for the 6D pose and can not be used in some

self-supervised methods, which require a fully differentiable network to transmit the signal between input and the 6D pose. Finally, the time consumed by the methods of PnP/RANSAC or voting and least square limits the real-time performance of the method when dealing with the dense correspondence.

To combine the advantages of the direct regression methods and the two-stage methods, [8], [17], [38] put forward direct regression frameworks based on the supervision of intermediate variables with RGB information for a better performance. However, lacking depth information restricts the ability to mine the geometric structure information of objects [3], which results in poor performance when suffering poor lighting conditions or low contrast and non-textured objects.

In this paper, we build an object pose estimator based on RGBD information. The RGBD full flow bidirectional fusion network in [3] is proven to be an efficient way to fuse the RGBD information, hence it is suitable for our backbone to utilize the texture and geometric structure information of the object. In the estimator, we integrate the end-to-end learning into the keypoint regression to realize gradient backpropagation between the 6D pose information and the RGBD information. Figure 1(c) illustrates an overview of our method. Using the RGBD full flow bidirectional fusion Network [3], we obtain the point-wise offset from the visible part of the object to the keypoints. Then, we remove the outliers by keypoint distribution. Specifically, we remain the points near the mean after standardizing the keypoints. The supervision of keypoints and the removal of outliers make the network more robust to learn geometric structure information of objects. Finally, we construct a simple but effective keypoint-based regression head. The head takes the keypoints in the camera coordinate system and the corresponding keypoints in the model coordinate system as input and outputs the 6D object pose. As the keypoints information comes from the previous features, the entire network is differentiable to the pose. Specially, we use the novel continuous representation for pose in [19] to improve the performance of learning. On account of these ways, we build an **E**nd-to-**E**nd regression network based on **K**eypoint (**E2EK**) for 6D pose estimation.

In addition, we propose a novel keypoint offset representation for our framework. Different from the previous work to supervise the keypoints offset [1], [3], we supervise the unit vector and length of offset. It is generally believed that uniformly distributed output is beneficial to the learning of the network [32]. Besides, we further improve points sampling rules in [1], [3]. The point cloud of the whole image is sampled randomly in [1], [3], which will lead to the limited amount of points on the small or heavily occluded objects. Hence we add an object detection framework and sample points around objects to ensure the number of object points for feature fusion.

To summarize, our main contributions are as follows.

1) We propose an end-to-end regression network based on the geometric information of correspondence keypoints, which makes the input differentiable to the 6D pose and improves the performance of 6D pose estimation.
2) We separate the keypoints offset into unit vector and length for regression to make the distribution of regression variables more uniform and improve the performance of keypoints estimation.
3) Extensive experiments on LM , LM-O , and YCBV datasets show the superiority of our method.

II. Related work

Traditional methods extract features manually and use template matching or point-pair feature matching to determine the pose of objects [22], [37]. Deep-learning methods have recently made satisfactory progress for the task. In this section, we review several different strategies for 6D object pose estimation.

Direct Method. The direct regression methods have the advantages of fast speed and differentiability for the 6D pose, but they lack accuracy. Early methods regard the pose estimation of objects as a discrete viewpoint classification problem [11], [18]. [11], [16] directly regress the 6D pose use the point-match loss. Specially, Densefusion [9] combines RGB and point cloud features to fully leverage the texture and geometric information. However, the above methods can only get rough results, so they use refinement to get better results. The reason why the above methods perform poorly is a) it is difficult for the network to encode relevant features only relying on the supervision of pose information b) the parameters used in most regression methods are quaternions or variants of quaternions [16], [19], which have been proved to be discontinuous in [19] and hence limits the learning of the network. [19] proposed a novel continuous expression for rotation in $SO(3)$, which further enhance the prospect of direct regression.

Nondifferentiable Indirect Method. The nondifferentiable indirect regression methods leverage the supervision of intermediate variables to mine the potential information of the pose. After that, the pose is estimated by a geometric estimator with the intermediate variables. Although they gain a higher accuracy, they are time-consuming and non-differentiable. Some methods predict dense 2D-3D correspondence, and then obtain the pose of the object through PnP/RANSAC [5], [6], [21], [41]. Specially, EPOS [6] uses fragments to divide the model into blocks and predict the category of pixels to solve the ambiguity problem of symmetrical objects. SurfEmb [41] proposes a novel query method based on key-value to solve the coordinate ambiguity problem of symmetric objects. CDPN [21] decouples the rotation and translation of the object, while the rotation matrix is obtained by PnP/RANSAC and the translation is obtained by regression. Some methods solve the pose depend on keypoints [1]–[3]. For example, PVNet [2] predicts the 2D offset in the pixel coordinate system from the visible part to the keypoints, and align the model after voting to estimate the pose. PVN3D [1] extends 2D to 3D to solve projection errors in 2D image.

Differentiable Indirect Method. To take advantage of the above methods, differentiable indirect methods construct differentiable networks while supervising the network using intermediate variables. Specially, [17] builds a network of point cloud structures to solve the 6D pose with the dense 2D-3D candidate correspondence. GDR-net [8] inputs the dense

2D-3D correspondence and object surface fragment prediction to a convolution network and full connection layer and output the 6D pose of the object. SO-Pose [40] introduces a new representation for self-occlusion and further improves the accuracy of object pose estimation. While they perform well, they lack accuracy when compared to the RGBD method. With RGBD information, G2L-Net [7] uses the offset from the visible points to the keypoints as supervision to obtain point-wise embedding features, and inputs these features into the fully-connected layer to obtain the 6D pose of the object. However, the unremoved outlier features may reduce the performance of the network and direct input features lack the keypoint information of the model. In the case of known object masks, REDE [12] designs a network to obtain the confidence of keypoints, and finally uses an end-to-end network to regress the 6D pose. However, when trained together with the task of mask prediction, the keypoint weight network tends to learn the same weights, resulting in poor performance. In addition, it requires a complex training strategy that solves the pose multi-times for single input during training.

III. Proposed method

Given an RGBD image captured by a calibrated camera, the goal of the 6D object pose estimation is to predict the transformation matrix between the object coordinate system and the camera coordinate system. The transformation matrix consists of the rotation matrix and a translation matrix. To tackle this problem better, the texture information of an RGB image and the geometric information of depth image should be used.

Fig. 2 shows an overview of our proposed method. We first detect the object of interest using an off-the-shelf object detector YOLOv3 [39] and sample the points in the bounding box as input to the Point Cloud Network(PCN). Based on the full flow bidirectional fusion network, we extract the point-wise RGBD feature for the semantic segmentation and the 3D keypoints localization. Combining the prediction results of semantic segmentation and keypoint offset, we get the set of keypoints. Then, we standardize it and average pool the points with high confidence in the Gaussian distribution to obtain the predicted keypoints. Through the correspondence between the predict keypoints and the model keypoints, we train a network for direct regression of object rotation and translation. The network consists of a $128 \times 128 \times 128$ 1-D convolution network and a full connection layer.

A. Point Sampling Module

As we use RGBD information, sufficient number of input point clouds can help obtain object structure information. We use the sampling principle in [1], [3] to randomly sample 12800 points from the entire image, and make the statistics in Fig. 3 on the Linemod Occulusion dataset.

It can be seen in Fig. 3 that when the object is occluded, the points randomly sampled on the object are very limited, although the overall sampling point has been as high as 12,800. On small objects such as cat, the sampling points on the surface of the object are less than 20 on 25% of the dataset. We think that too sparse points are difficult to represent the geometric structure of the object. Therefore, we added an offline object detection framework. It is worth noting that since our object detection framework is independent of the following pose estimation network, we can directly use methods with both fast and high-accuracy in the rapidly growing field of object detection. Meanwhile, sampling in the vicinity of the object can significantly reduce the sampling point, thereby reducing calculation memory and time consumption.

B. Keypoints Learning Module

Considering that [3] is the state-of-the-art network for rgb and point cloud fusion, we choose it as the feature extractor. Following [3], we use ImageNet pre-trained ResNet34 to encode RGB images and the PSPNet [28] as a decoder. For the point cloud processing, we apply the RandLA-Net [29] following [3] for a fair comparison. After the full flow bidirectional fusion networks, the dense RGBD features are obtained and fed into the semantic segmentation and the keypoint offset learning modules.

1) semantic segmentation module: After extracting the per-point features, we use the semantic segmentation module composed of shared MLP layers to predict the per-point semantic labels. The main function of semantic segmentation is to segment object points and non-object points (background or other objects) in the object detection boundingbox. In the following keypoint voting module, we only take points judged as objects. We use Focal Loss [31] as the loss function of the module.

$$L_{semantic} = -\delta(1-q_i)^{\eta} log(q_i) \tag{1}$$

where $q_i = c_i * l_i$, c_i presents the predicted confidence for the ith point belongs to each class, l_i is the one-hot representation of the ground true class label, δ is the balance parameter and η is the focusing parameter.

2) 3D keypoints offset detection module: Similar to [1]–[3], we predict per-point euclidean translation offset from the object visible points to target keypoints. Different from [1] to predict the unit vector, or [2] to predict the offset with length, we predict the unit vector and the length of the offset. In other words, we predict the four-dimensional vectors, including the direction and the length of the offset. We think the uniform distribution of the unit vector is helpful to the learning of the network, but it loses the length information, which may bring some additional errors to the recovery of keypoints after network learning.

Given a set of visible seed points $\{p_i\}_{i=1}^{N}$ and a set of selected keypoints $\{k_j\}_{j=1}^{M}$ belongs to an object, we predict the unit offset $\{v_i^j\}$ and the length of the offset $\{l_i^j\}$, where i presents the ith visible points and j presents the jth keypoints. In other words, for the visible points p_i of the instance, the jth candidate keypoint is define as $k_j = p_i + v_i^j * l_i^j$. We use L1 Loss to supervise the offset and the length.

$$\begin{cases} L_{keypoints} = L_{unit} + L_{length} \\ L_{unit} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}\|v_i^j - v_i^{j*}\|_1 \Gamma(p_i \in I) \\ L_{length} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}\|l_i^j - l_i^{j*}\|_1 \Gamma(p_i \in I) \end{cases} \tag{2}$$

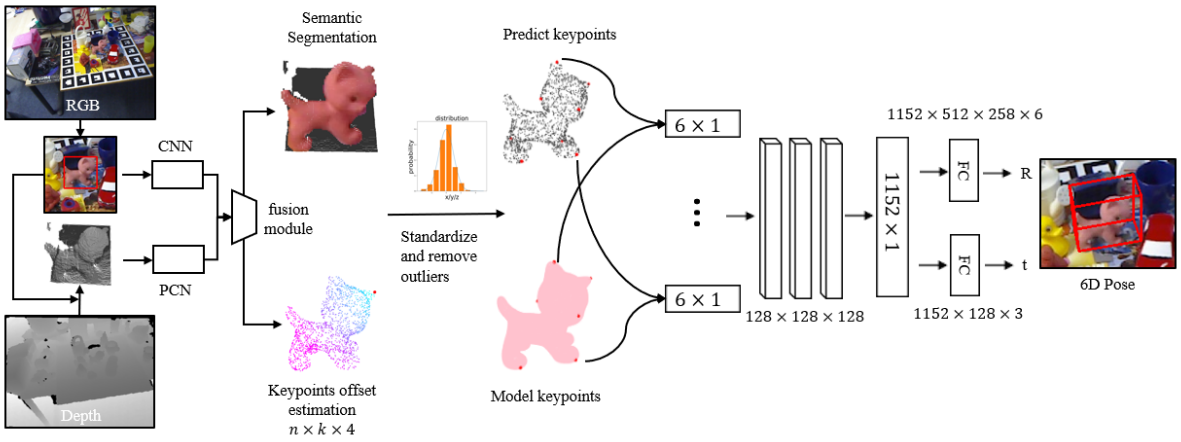


Fig. 2. Overview of the proposed method. To ensure a sufficient number of sampling points on the object, we first detect the object and sample around the object. After that, we leverage the full flow bidirectional fusion network to fuse RGB features and point cloud features to supervise keypoints offset. After voting keypoints, the high-confidence keypoints and the model keypoints are concatenated, and the 6D pose of the object is finally obtained after the keypoint-base network.

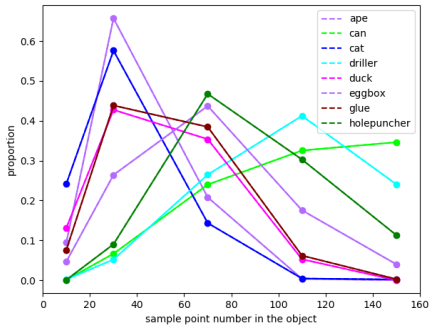


Fig. 3. Proportion of sampled points on the surface of the object. The abscissa is the range of sampling points, and the ordinate is the proportion of data in the entire dataset.

where the v_i^{j*} is the ground truth translation unit vector offset; N is the total number of the points and M is the number of the keypoints. Γ is an indicating function. When p_i belongs to instance I, it equals to 1 otherwise 0.

C. Regression Module

We propose a keypoints-guided network to replace clustering and least-squares in [1], [3]. This method can make the network end-to-end trainable, and the loss function is directly related to the accuracy of object pose estimation. Firstly, we use the network to predict the points of each object and the offset vector to the keypoints to obtain the candidate keypoints of the object. Then, the keypoints are standardized, and the keypoints with high confidence are selected for average pooling to obtain the predicted keypoints. Finally, we concatenate the predicted keypoint and the corresponding keypoint in the model coordinate system, which is passed through 1-D convolutional networks and fully connected layers to obtain the pose of the object.

1) keypoints voting: For keypoints $\{{}^c k_j\}_{j=1}^m$ in camera coordinate system, we get the set of candidate points $\{\{p_j^i\}_{i=1}^n\}_{j=1}^m$, where m is the number of the keypoints and n is the numbers of object points judged from the semantic segmentation. For each keypoint ${}^c k_j$, we standardize the point set $\{p_j^i\}_{i=1}^n$ to evaluate the per-point confidence and obtain the keypoints after voting.

$$ {}^c k_j = \sum_{i=1}^{n} (p_j^i * c_j^i) / \sum_{i=1}^{n} (c_j^i) \tag{3} $$

where c_j^i represents the confidence of keypoints, which can be calculated by the following equation.

$$ {}^s p_j^i = (p_j^i - mean(\{p_j^i\}_{i=1}^n)) / std(\{p_j^i\}_{i=1}^n) \tag{4} $$

$$ c_j^i = \begin{cases} 1 & |{}^s p_j^i| < \theta \\ 0 & |{}^s p_j^i| > \theta \end{cases} \tag{5} $$

where $mean$ and std represent the mean and standard deviation of the point set $\{p_j^i\}_{i=1}^n$, respectively. θ is a threshold and we find $\theta = 0.6$ to be optimal.

2) pose regression: To enable the network to better learn the correspondence between keypoints, we aggregate model keypoints and predicted keypoints as the input of the regression network. In conclusion, our regression module can be written as follow.

$$ (R, t) = G(cat\{({}^c k_1, {}^m k_1), ..., ({}^c k_j, {}^m k_j)\}, \Theta) \tag{6} $$

where ${}^m k_j$ and ${}^c k_j$ represent the keypoints in the model system and camera system respectively. $cat()$ represent the concatenation operation. For function G, we use a one-dimensional convolutional network followed by two fully connected layers to realize. It is worth noting that although the network structure looks simple, the keypoint information comes from the previous features filtered by voting, so the gradient can be returned to the original RGBD information. Meanwhile, multi-task training improves the ability of the network to extract features.

TABLE I
EXPERIMENT ON LINEMOD DATASET

	RGB				RGBD					
	PoseCNN [25]	PVNet [2]	CDPN [21]	DPOD [5]	PointFusion [26]	Dense-fusion [9]	G2L-Net [7]	PVN3D [1]	FFB6D [3]	Ours
ape	77	43.6	64.4	87.7	70.4	92.3	96.8	97.3	98.4	**98.7**
benchvise	97.5	99.9	97.8	98.5	80.7	93.2	96.1	99.7	100	100
camera	93.5	86.9	91.7	96.1	60.8	94.4	98.2	99.6	99.9	99.9
can	96.5	95.5	95.9	99.7	61.1	93.1	98	99.5	99.8	**100**
cat	82.1	79.3	83.8	94.7	79.1	96.5	99.2	99.8	99.9	**100**
driller	95	96.4	96.2	98.8	47.3	87	99.8	99.3	100	100
duck	77.7	52.6	66.8	86.3	63	92.3	97.7	98.2	98.4	**99.4**
eggbox	97.1	99.2	99.7	99.9	99.9	99.8	100	99.8	100	100
glue	99.4	95.7	99.6	96.8	99.3	100	100	100	100	100
holepuncher	52.8	82	85.8	86.9	71.8	92.1	99	99.9	99.8	**100**
iron	98.3	98.9	97.9	100	83.2	97	99.3	99.7	99.9	**100**
lamp	97.5	99.3	97.9	96.8	62.3	95.3	99.5	99.8	99.9	99.9
phone	87.7	92.4	90.8	94.7	78.8	92.8	98.9	99.5	99.7	**100**
mean	88.6	86.3	89.9	95.1	73.7	94.3	98.7	99.4	99.7	**99.8**

* Italic and bold are symmetrical objects

TABLE II
EXPERIMENT ON LINEMOD OCCULUSION DATASET.

	PoseCNN [25]	Pix2Pose [4]	PVNet [2]	DPOD [5]	HybridPose [27]	Single-stage [17]	PVN3D [1]	FFB6D [3]	Ours
ape	9.6	22.0	15.8	-	20.9	19.2	33.9	47.2	**61.0**
can	45.2	44.7	63.3	-	75.3	65.1	88.6	85.2	**95.4**
cat	0.9	22.7	16.7	-	24.9	18.9	39.1	45.7	**50.8**
driller	41.4	44.7	65.7	-	70.2	69.0	78.4	81.4	**94.5**
duck	19.6	15.0	25.2	-	27.9	25.3	41.9	53.9	**59.6**
eggbox	22.0	25.2	52.0	-	52.4	52.0	**80.9**	70.2	55.7
glue	38.5	32.4	51.4	-	53.8	51.4	68.1	60.1	**78.3**
holepuncher	22.1	49.5	45.6	-	54.2	45.6	74.7	85.9	**91.4**
mean	24.9	32.0	40.8	47.3	47.5	47.6	63.2	66.2	**73.3**

* Italic and bold are symmetrical objects

Following [8], [19], we use the 6D representation R_{6d} in $SO(3)$, which is continuous in Euclidean space and conducive to network learning. Specifically, R_{6d} is consists of the first two columns of rotation matrix R.

$$R_{6d} = [R_1|R_2] \tag{7}$$

The rotation matrix R can be computed by the prediction $R_{6d} = [r1|r2]$ according to

$$\begin{cases} R_1 = \phi(r_1) \\ R_3 = \phi(R_1 \times r_2) \\ R_2 = R_3 \times R_1 \end{cases} \tag{8}$$

where ϕ represents the vector cross product operation. Finally, the pose loss is defined as follow, where μ is the balance parameter and we set to 2.0 in our experiment.

$$\begin{cases} L_{pose} = L_R + \mu L_t \\ L_R = avg_{x \in M} \|\hat{R}x - \bar{R}x\| \\ L_t = \|\hat{t} - \bar{t}\| \end{cases} \tag{9}$$

For the symmetric objects,we compute the set Ω including all possible ground-truth rotations under symmetry and then compute the loss as $L_{R,sym} = \min_{\bar{R} \in \Omega} L_R(\hat{R}, \bar{R})$.

Thereby, we use the following loss to guide the training of the network.

$$L = \alpha L_{semantic} + \beta L_{keypoints} + \gamma L_{pose} \tag{10}$$

IV. EXPERIMENT

In this section, we first introduce our experimental setup and implementation details and then report the evaluation results for several commonly employed benchmark datasets. Specifically, we compared our method with the start-of-the-art methods on LM, LM-O and YCB-V. Finally, we demonstrate the effectiveness of our individual components by performing an ablative study on Linemod Occulusion Dataset.

A. Datasets

LINEMOD [22] includes 13 sequences of 13 objects with pose annotation, each with about 1200 images. It contains chaotic scenes and changeable lighting. We follow the previous work to divide the data set, use about 15% of the pictures for training [1]–[3], the remaining pictures for testing, and follow [1]–[3] to generate synthetic images for expanding the data set.

LINEMOD OCCULUSION [23] is a subsequence belonging to LM, which consists of 1214 images with 8 heavily occluded objects pose annotations. We use it only for the test following [1]–[3].

YCB-V [24] contains 92 RGBD image sequences, with a total of more than 110k images. It includes a total of 21 objects, including symmetrical objects and low texture objects. We follow [1], [3] to divide the training set and the test set. In addition, we follow [1], [3] and use the synthetic data set

in [25] to expand the training set, and use the hole filling algorithm to fill the depth map of the synthetic data set [3].

B. Implementation Details

All our experiments are implemented using PyTorch. We train the model for a total of 40 epochs. For the learning rate, we choose the CyclicLR [35] while the max learning rate is set to $1e{-}3$ and the base learning rate is set to $1e{-}5$. Specifically, we use 2 cycles for Linemod and Linemod Occulusion dataset and 6 cycles for YCB-V datasets. For the hyperparameter in Eq. (10), we set $\alpha = 2.0, \beta = 1.0, \gamma = 0.001$ at first 20 epochs to let the network better learn semantics and keypoints, and improve the γ to 2.0 at end 20 epochs to let the network learn the object pose information.

C. Evaluation Metrics

Following [1], [3], we use the average distance metrics ADD and ADDS for evaluation. ADDS is generally used to evaluate symmetrical objects. ADD and ADDS is defined as follows:

$$ADD = \frac{1}{m} \sum_{v \in O} \|(Rv + t) - (R^* v + t^*)\|$$

$$ADDS = \frac{1}{m} \sum_{v_1 \in O} \min_{v_2 \in O} \|(Rv_1 + t) - (R^* v_2 + t^*)\|$$

where v denotes a point in object, R, t is the predicted pose and the R^*, t^* is the ground truth.

Following previous methods, we report $ADD(S) < 0.1d$ on the LM and LM-O, where d means the diameter of the object and ADD(S) means using ADD for non-symmetric objects while ADDS for symmetric objects. When evaluating on YCB-V, we compute the AUC of the ADDS and ADD(S), representing the area under the ADDS and ADD(S) curve.

D. Evaluation on Three Benchmark Datasets

Evaluation on the Linemod Dataset. Table I shows our results. Compared to state-of-art methods, we achieved the best performance on every object and get an average of 99.8% on $ADD(S) < 0.1d$. The testing set of the LM is very close to the training set in the scene, and the objects are not occluded, which leads to the performance of other methods being good enough. Thus our method has a limited improvement in comparison.

Evaluation on the Linemod Occulusion Dataset. To verify the performance of our method under occlusion, we conducted experiments on LM-O. We follow the state-of-the-art to train our model on the LM dataset and only use this dataset for testing. Table II demonstrates our results. Compared to the start-of-art method, we improve a lot in most of the objects except the eggbox. We found that the eggbox lacks texture and it has a poor performance in semantic segmentation. if using the ground truth, we also get the 84.2% $ADDS < 0.1d$ on the eggbox.

Evaluation on the YCB Dataset. Table III shows our results. Compared to the latest method, we achieved 94.4% on ADD(S) AUC, which is 0.7% higher than start-of-the-art methods.

E. Ablation Study

Ablation on Voting strategies. We conduct a comparison experiment between the mean-shift and our standardization. Then benchmark is the model only with the object detection. We first train the model with keypoints offset, the two different voting strategies are applied respectively to get the input of Least Squares. We conduct the experiment on the LM-O. From Table IV, our method achieves the same performance with lower time consumption. Specifically, in the keypoints error, our method achieves better performance with the slight advantage of 1mm. But in terms of time consumption, we have achieved a great advantage. The time complexity for the mean-shift is $O(n^2)$ while standardization is $O(n)$. It is worth noting that the speed here is slower than that in [2], [3]. The reason is that we use object detection to sample points around the object, which causes the points on the object to be much denser.

Ablation on Regression network. In this experiment, we compare the results between the geometric estimator using different voting strategies and the direct regression. We evaluate our models in the LM-O. Table V shows our results. Experiments show that our voting strategy performance is not much different from mean-shift. Meanwhile, the performance of using the keypoint regression network is better. This shows that our proposed network can better learn the relationship between point pairs.

Ablation on our method. In this experiment, we respectively compared the results of the individual improvements we proposed. We evaluate our models in the LM-O. Table VI shows our results. Because the selecting keypoints of symmetrical objects according to [3] are ambiguous, and our method is based on keypoints, we omit symmetrical objects in this ablation for better comparison. Experiments show that the individual improvements we proposed enhance the performance. Specifically, the small object such as ‘ape‘ benefit a lot from increasing the sampled points of the object, increasing by 11%. Large objects benefit from the unitized loss because the unnormalized offset vectors of large objects have a larger distribution range. Among them, ‘driller‘ and ‘can‘ improve by 12.7% and 9.3% respectively.

F. Running Time

On the desktop of Intel(R) i7 @3.10GHz CPU and the RTX 3090 GPU, given 480×680 RGB and depth image, our method takes approximately 68ms for a single object, for which 8ms for the object detection.

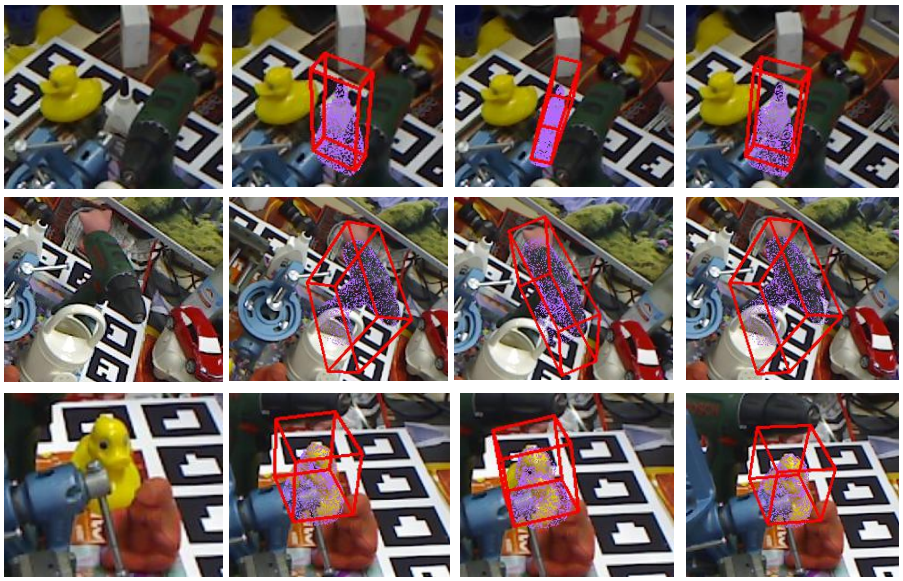
V. CONCLUSIONS

In this paper, we integrate a keypoint-based regression model to the 6D pose estimation task. Hence, end-to-end learning is achieved by using the intermediate variable. Meanwhile, we improve the sampling method for point cloud, which benefit the small objects. Furthermore, a new unit loss consisting of a unit vector and the length is proposed to improve the keypoints learning. Experiments show our methods outperform other methods. However, our method has some shortcomings in dealing with symmetric objects, because

TABLE III
EXPERIMENT ON THE YCB-V DATASET.

	PoseCNN [25]		DCF [30]		Densefusion [9]		PVN3D [1]		FFB6D [3]		Ours	
Object	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)	ADDS	ADD(S)
002	83.9	50.2	90.9	74.6	95.3	70.7	96	80.5	**96.3**	**80.6**	95.3	79.6
003	76.9	53.1	87.1	79.3	92.5	86.9	96.1	94.8	96.3	94.6	**96.4**	**95.1**
004	84.2	68.4	94.3	84.2	95.1	90.8	97.4	96.3	97.6	96.6	**97.7**	**96.7**
005	81.0	66.2	90.5	79.8	93.8	84.7	96.2	88.5	95.6	89.6	**95.6**	**89.8**
006	90.4	81.0	90.6	83.5	95.8	90.9	97.5	96.2	**97.8**	**97.0**	97.5	96.5
007	88.0	70.7	91.7	73.8	95.7	79.6	96	89.3	96.8	88.9	**97.5**	**90.7**
008	79.1	62.7	89.3	84.1	94.3	89.3	97.1	95.7	97.1	94.6	**97.8**	**96.9**
009	87.2	75.2	92.9	89.5	97.2	95.8	97.7	96.1	98.1	96.9	**98.3**	**97.5**
010	78.5	59.5	83.2	74.6	89.3	79.6	93.3	88.6	**94.7**	88.1	93.3	**90.8**
011	86.0	72.3	84.8	71.0	90.0	76.7	96.6	93.7	97.2	**94.9**	**97.4**	94.4
019	77.0	53.3	89.5	80.3	93.6	87.1	97.4	96.5	**97.6**	**96.9**	97.0	95.6
021	71.6	50.3	88.4	79.8	94.4	87.5	96.0	93.2	**96.8**	**94.8**	96.7	94.0
024	69.6	69.6	80.3	80.3	86.0	86.0	90.2	90.2	**96.3**	**96.3**	96.0	96.0
025	78.2	58.5	90.7	76.6	95.3	83.8	97.6	95.4	97.3	94.2	**97.3**	**95.3**
035	72.7	55.3	87.4	78.4	92.1	83.7	96.7	95.1	97.2	95.9	**97.4**	**96.6**
036	64.3	64.3	84.2	84.2	89.5	89.5	90.4	90.4	92.6	92.6	**93.8**	**93.8**
037	56.9	35.8	84.2	70.3	90.1	77.4	96.7	92.7	97.7	95.7	**98.5**	**97.9**
040	71.7	58.3	89.5	81	95.1	89.1	96.7	91.8	96.6	89.1	**97.8**	**95.0**
051	50.2	50.2	63.6	63.6	71.5	71.5	93.6	93.6	96.8	96.8	**97.2**	**97.2**
052	44.1	44.1	64.4	64.4	70.2	70.2	88.4	88.4	96.0	96.0	**96.7**	**96.7**
061	88	88	83.1	83.1	92.2	92.2	96.8	96.8	**97.3**	**97.3**	97.2	97.2
mean	75.2	61.3	85.7	77.9	90.9	84.0	95.4	92.6	96.6	93.7	**96.8**	**94.4**

* Italic and bold are symmetrical objects



(a) image (b) ground truth (c) FFB6D (d) Ours

Fig. 4. Comparison result on Linemod Occulusion dataset.

TABLE IV
ABLATION ON VOTING STRATEGIES

	keypoints error/cm	ADD(S) <0.1d	time /ms
standardization	3.4	67.9	5
meanshift	3.5	67.7	83

TABLE V
ABLATION ON REGRESSION NETWORK

	Meanshift	Standardization	Regression
ADD(S)<0.1d	72.7	72.9	73.3
ADDS<0.1d	83.9	83.8	84.3

TABLE VI
ABLATION ON OUR PROPOSE METHODS.

object detection		✓			✓	✓		✓
unit loss			✓		✓		✓	✓
regression head				✓		✓	✓	✓
ape	47.2	58.2	53.6	49.6	54.9	59.4	59.5	61.0
can	85.2	90.4	94.5	85.5	94.9	92.4	92.8	95.4
cat	45.7	46.3	50.6	46.0	48.4	45.0	47.0	50.8
driller	81.4	90.3	94.1	90.8	91.6	92.1	94.3	94.5
duck	53.9	59.0	55.2	57.4	57.1	60.0	60.3	59.6
holepuncher	85.9	75.9	80.3	86.2	87.2	87.8	84.0	91.4
mean	66.6	70.0	71.4	69.3	72.3	72.8	73.0	75.5

the keypoint definitions are ambiguous for symmetric objects.

In the future, we will conduct a more detailed study on this problem. In addition, we will also consider the advantages of end-to-end training and use it for self-supervised learning.

REFERENCES

[1] Y. He, W. Sun, H. Huang, *et al*. "Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2020,pp. 11632-11641.

[2] S. Peng, Y. Liu, Q. Huang, *et al*. "Pvnet: Pixel-wise voting network for 6dof pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2019, pp. 4561-4570.

[3] Y. He, H. Huang, H. Fan, *et al*. "Ffb6d: A full flow bidirectional fusion network for 6d pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2021, pp. 3003-3013.

[4] K. Park , T. Patten , M. Vincze . "Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation." " in *Proc. Int. Conf. Comput. Vis.,* 2019, pp. 7668-7677.

[5] S. Zakharov , I. Shugurov, S. Ilic. "Dpod: 6d pose object detector and refiner." in *Proc. Int. Conf. Comput. Vis.,* 2019, pp.1941-1950.

[6] T. Hodan, D. Barath, J. Matas. "Epos: Estimating 6d pose of objects with symmetries." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2020,pp. 11703-11712.

[7] W. Chen, *et al*. "G2l-net: Global to local network for real-time 6d pose estimation with embedding vector features." in *Proc. Int. Conf. Comput. Vis.,* 2020, pp. 4233-4242.

[8] G. Wang, F. Manhardt , F. Tombari, *et al*. "Gdr-net: Geometry-guided direct regression network for monocular 6d object pose estimation." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2021, pp. 16611-16621.

[9] C. Wang, D. Xu, Y. Zhu, *et al*. "Densefusion: 6d object pose estimation by iterative dense fusion." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2019, pp. 3343-3352.

[10] Lepetit, Vincent, and F. Pascal . Monocular model-based 3D tracking of rigid objects. *Now Publishers Inc*, 2005.

[11] W. Kehl, F. Manhardt, F. Tombari, *et al*. "Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again." in *Proc. Int. Conf. Comput. Vis.,* 2017, pp. 1521-1529.

[12] W. Hua, Z. Zhou, J. Wu, *et al*. "Rede: End-to-end object 6d pose robust estimation using differentiable outliers elimination." *IEEE Robotics and Automation Letters,* vol. 6, no. 2, pp. 2886–2893, 2021.

[13] G. Wang, F. Manhardt, J. Shao, *et al*. "Self6d: Self-supervised monocular 6d object pose estimation." in *Proc. Eur. Conf. Comput. Vis.,* 2020, pp. 108-125.

[14] D. Beker, H. Kato, M. Morariu, *et al.* "Monocular differentiable rendering for self-supervised 3d object detection." in *Proc. Eur. Conf. Comput. Vis.,* 2020, pp. 514-529.

[15] Y. Labbé, J. Carpentier , M. Aubry, *et al*. "Cosypose: Consistent multi-view multi-object 6d pose estimation." in *Proc. Eur. Conf. Comput. Vis.,* 2020, pp. 574-591.

[16] Y. Li, G. Wang, X. Ji, *et al*. "Deepim: Deep iterative matching for 6d pose estimation." in *Proc. Eur. Conf. Comput. Vis.,* 2018, pp. 683-698.

[17] Y. Hu, P. Fua , W. Wang, *et al*. "Single-stage 6d object pose estimation." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2020, pp. 2930-2939.

[18] J. Papon, M. Schoeler. "Semantic pose using deep networks trained on synthetic RGB-D." in *Proc. Int. Conf. Comput. Vis.,* 2015, pp. 774-782.

[19] Y. Zhou, C. Barnes, J. Lu, *et al*. "On the continuity of rotation representations in neural networks." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2019, pp. 5745-5753.

[20] K. Park, A. Mousavian, Y. Xiang, *et al*."Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2020, pp. 10710-10719.

[21] Z. Li, G. Wang G, X. Ji. "Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation." in *Proc. Int. Conf. Comput. Vis.,* 2019, pp. 7678-7687.

[22] S. Hinterstoisser, S. Holzer, C. Cagniart, *et al*. "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes." in *Proc. Int. Conf. Comput. Vis.,* 2011, pp. 858-865.

[23] E. Brachmann, A. Krull, F. Michel, *et al*. "Learning 6d object pose estimation using 3d object coordinates." in *Proc. Eur. Conf. Comput. Vis.,* 2014, pp. 536-551.

[24] B. Calli, A. Singh, A. Walsman, *et al*. "The ycb object and model set: Towards common benchmarks for manipulation research." in *international conference on advanced robotics,* 2015, pp. 510-517.

[25] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6 d object pose estimation in cluttered scenes," *Robot.: Sci. Syst.,* 2018, doi: 10.15607/RSS.2018.XIV.019.

[26] D. Xu, D. Anguelov, A. Jain, "Pointfusion: Deep sensor fusion for 3d bounding box estimation." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2018, pp. 244-253.

[27] C. Song, J. Song, Q. Huang, "Hybridpose: 6d object pose estimation under hybrid representations." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2020. pp. 431-440.

[28] H. Zhao, J. Shi, X. Qi , *et al.*, "Pyramid scene parsing network." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2017. pp. 2881-2890.

[29] Q. Hu, B. Yang, L. Xie, *et al*.,"Randla-net: Efficient semantic segmentation of large-scale point clouds." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2020. pp. 11108-11117.

[30] M. Liang, B. Yang, S. Wang, *et al.*, "Deep continuous fusion for multi-sensor 3d object detection." in *Proc. Eur. Conf. Comput. Vis.,* 2018, pp. 641-656.

[31] Y. Lin, P. Goyal, R. Girshick, *et al.*,"Focal loss for dense object detection." in *Proc. Int. Conf. Comput. Vis.,* 2017, pp. 2980-2988.

[32] S. Ioffe, C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift." in *International conference on machine learning,* 2015, pp. 448-456.

[33] H. Lin, T. Zhang, Z. Chen, *et al.*, "Adaptive fuzzy Gaussian mixture models for shape approximation in robot grasping." *International Journal of Fuzzy Systems* vol. 21, no. 4, pp. 1026–1037, 2019.

[34] Z. Lu, N. Wang, C. Yang, "A novel iterative identification based on the optimised topology for common state monitoring in wireless sensor networks." *International Journal of Systems Science* vol. 53, no. 1, pp. 25-39, 2022.

[35] Smith, N. Leslie. "Cyclical learning rates for training neural networks." in *IEEE winter conference on applications of computer vision,* 2017, pp. 464-472.

[36] C. Li,, B. Jin, and H. Gregory D, "A unified framework for multi-view multi-class object pose estimation." in *Proc. Eur. Conf. Comput. Vis.,* 2018, pp. 254-269.

[37] J. Bohg, A. Morales, T. Asfour, *et al.*, "Data-driven grasp synthesis—a survey." *IEEE Transactions on robotics* vol. 30, no. 2, pp. 289-309, 2013.

[38] G. Billings, and M. Johnson-Roberson, "Silhonet: An rgb method for 6d object pose estimation." *IEEE Robotics and Automation Letters,* vol. 4, no. 4, pp. 3727-3734, 2019.

[39] J. Redmon , A. Farhadi , "Yolov3: An incremental improvement, " 2018, *arXiv:1804.02767*.

[40] Y. Di, F. Manhardt , G. Wang , *et al.*, "SO-Pose: Exploiting Self-Occlusion for Direct 6D Pose Estimation." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2021. pp. 12396-12405.

[41] Haugaard, et al. ,"SurfEmb: Dense and Continuous Correspondence Distributions for Object Pose Estimation with Learnt Surface Embeddings." 2021, *arXiv:2111.13489*.

[42] Du, Guoguang, et al. "Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review." *Artificial Intelligence Review,* vol. 54, no. 3, pp. 1677-1734,2021.

[43] E. Marchand, H. Uchiyama, F. Spindler, "Pose estimation for augmented reality: a hands-on survey." *IEEE transactions on visualization and computer graphics* vol. 22, no. 12,pp. 2633-2651, 2015.

[44] F. Manhardt, W. Kehl, A. Gaidon, "Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape." in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.,* 2019. pp. 2069-2078.

[45] M. Gonzalez, A. Kacete, A. Murienne, *et al.*, "L6dnet: Light 6 dof network for robust and precise object pose estimation with small datasets." *IEEE Robotics and Automation Letters,* vol. 6, no. 2, pp. 2914-2921, 2021.