2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Joint-Loss Enhanced Self-Supervised Learning for Refinement-Coupled Object 6D Pose Estimation

Fengjun Mu[1*], Shixiang Sun[1*], Rui Huang[1†], Chaobin Zou[1], Wenjiang Li[1], Huayi Zhan[2] and Hong Cheng[1]

***Abstract*— 6D object pose estimation plays a crucial role in robot grasping and manipulation. However, the prevalent methods for 6D object pose estimation heavily rely on 6D annotated data to train deep neural networks, which poses challenges due to the difficulty in obtaining sufficient pose annotations. To address this limitation, this paper presents a self-supervised pose estimation method based on a novel pixel-wise weighted dense fusion architecture. This method allows for direct learning from unannotated RGB-D data facilitated by an Iterative Annotation Resolver. Furthermore, a self-supervised pose refinement method based on joint loss is proposed to enhance the pose estimation accuracy. This refinement method employs a differentiable renderer to construct joint optimization constraints. The experimental results demonstrate that our approach achieves a level of pose estimation accuracy that closely rivals that of supervised methods.**

I. INTRODUCTION

6D object pose estimation holds great potential in various domains such as robotic grasping [1][2], autonomous driving [3][4], and augmented reality [5][6]. In recent years, owing to advancements in computer vision technology and the widespread availability of affordable RGB-D cameras, using RGB-D data for object pose estimation has gained popularity. Nevertheless, compared to the ease of acquiring RGB-D data, annotating object poses in large-scale RGB-D datasets remains an expensive challenging task. The scarcity of annotated object poses constitutes a significant obstacle to the practical implementation of object pose estimation. Therefore, researching how to leverage unannotated RGB-D data to achieve practical accuracy in object pose estimation is an immensely valuable approach.

Traditional pose estimation methods rely on manual feature extraction from 2D images and 3D point clouds, such as SIFT [7], FAST [8], FPFH [9], and CVFH [10]. These methods establish correspondences with 3D models to recover the object pose. However, manual feature-based approaches demonstrate limited robustness in scenarios involving occlusion and lighting variations. Recently, deep neural networks have made significant strides in pose estimation using RGB-D data [11][12][13]. Nonetheless, these deep learning methods heavily rely on high-precision ground truth pose annotations, which limits the practical application of object pose estimation.

[1]Fengjun Mu, Shixiang Sun, Rui Huang, Chaobin Zou, Wenjiang Li and Hong Cheng are with the Center for Robotics, School of Automation Engineering, University of Electronic Science and Technology of China, 611731 Chengdu, China

[2]Huayi Zhan is with the Changhong AI Research (CHAIR), Sichuan Changhong Electronic (Group) Co., Ltd., 621000 Mianyang, China

*Contributed equally to this work.

†Corresponding author: Rui Huang ruihuang@uestc.edu.cn

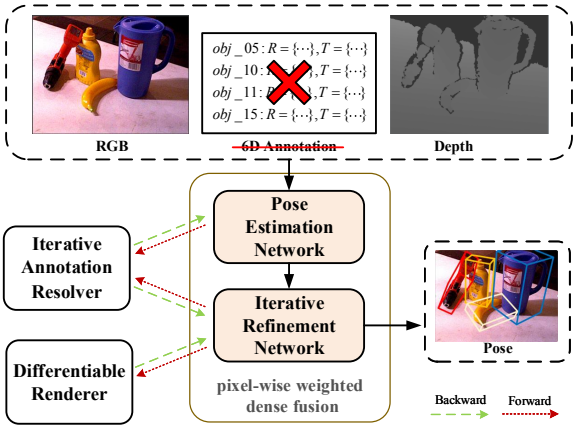

Fig. 1. We propose a pose estimation network along with an iterative refinement network based on a pixel-wise weighted dense fusion architecture. These networks are optimized using self-supervised loss functions generated by an Iterative Annotation Resolver and a differentiable renderer, allowing them to learn from unannotated RGB-D data.

To enhance the accuracy of pose estimation, iterative refinement techniques are commonly employed. Traditional methods such as ICP [14] and Colored-ICP [15] are used for 3D point cloud registration. However, these methods heavily rely on the initial object pose, exhibit limited refinement accuracy, and can be computationally intensive. Recently, pose refinement methods based on deep learning [11][16][17] have shown promising results. These approaches can effectively improve pose estimation accuracy without time-consuming iterative optimization. Nevertheless, these deep learning-based refinement methods still require object pose annotations for training supervision.

This paper presents a novel approach for pose estimation and refinement, leveraging unannotated RGB-D data for learning. The proposed approach is assessed in the popular benchmark YCB-Video dataset. Compared to existing methods, our approach eliminates the need for expensive high-precision pose annotation while achieving estimation results that are close to those achieved through supervised training.

In summary, the main contributions of this paper are:

- We propose a novel pose estimation model that leverages a pixel-wise weighted dense fusion architecture, which can enhance the efficiency of self-supervised learning.
- We propose a self-supervised pose refinement method, which constructs a self-supervised joint loss function through a differentiable renderer, enabling learning from unannotated data.

- Quantitative and qualitative experiments were evaluated on the YCB-Video dataset. The results demonstrate the effectiveness of our method.

II. RELATED WORK

A. Supervised Pose Estimation

6D object pose estimation can be classified into three categories. **Correspondence-based methods** aim to establish the correspondence between the 3D model and input data, and then use algorithms such as RANSAC [18] and PnP [19] to recover the pose. Traditional approaches rely on manual descriptors to extract features from 2D images [7][8][20][21] and 3D point clouds [9][10][22]. However, these manual feature methods often suffer from poor robustness in scenarios involving changes in lighting and occlusion. With the development of deep learning, an increasing number of studies are utilizing deep neural networks, such as CNNs, to extract features. For example, 3DMatch [23] facilitates the establishment of matching relationships between 3D models using voxel-based deep neural networks. **Template-based methods** establish a feature template database and match the input data features with the template to recover the object pose [24][25][26][27][28]. For example, CDPN [24] operates on 2D images and predicts rotation and translation independently by decoupling the pose. Similarly, G2L-Net [26] and Gao et al. [27] use local point cloud regression to estimate the object's pose. **Voting-based methods** involve predicting multiple poses for each input pixel or 3D coordinate point and then determining the final estimated pose through voting. This approach is particularly useful for pose estimation of occluded objects. DenseFusion [11] introduces a heterogeneous network that processes input 2D images and 3D point clouds. It densely fuses image and geometric features to generate object poses and confidences for each pixel, selecting the final estimated pose based on the confidences. PVNet [29] and PVN3D [30] employ voting on key points to find 2D-3D correspondences for pose calculation. It's worth noting that the aforementioned supervised pose estimation methods reliance on extensive pose annotation for establishing correspondence, constructing template matching library, and supervising the pose voting process.

B. Self-Supervised Pose Estimation

In the field of 6D object pose estimation, several approaches aim to eliminate the manual annotation process by using annotated synthetic data. Self6D [31] employs a differentiable renderer to generate images from the object model and estimated pose. It aligns the rendered image with the input image to construct a self-supervised loss function. Self6D++ [32] uses synthetic data to supervise the training of teacher models, and subsequently trains the noisy student model using unannotated real-scene data. Zhou [33] adopts a strategy of training the network by taking the mixed synthetic and real-scene data as input in a single bath, aiming to diminish the domain gap between synthetic and real-scene data. Weak6D [34] introduces an Iterative Annotation Resolver to generate Inaccurate learning objectives, directly utilizing unannotated RGB-D data to construct constraint conditions.

C. Pose Refinement Method

Pose refinement is commonly employed to enhance the accuracy of pose estimation. The traditional method ICP [14] relies on point cloud coordinates for iterative refinement, without utilizing object texture information. Colored-ICP [15] extends ICP by incorporating color constraints. In recent years, deep learning-based refinement methods [29][35][36] have emerged. These methods iteratively train networks to achieve the ability of pose refinement. For example, DeepIM [16] and Trabelsi et al. [17] use monocular images as input and leverage FlowNet to iteratively refine pose through matching rendered and observed images. Lipson et al. [36] refine the pose and correspondence iteratively in a tightly coupled manner. While these deep learning iterative refinement methods can significantly improve the accuracy of pose estimation, it is important to note that they still rely on high-precision pose annotation for training.

III. METHODS

A. Overview

Our method contains two stages: pose estimation and pose refinement. Both stages employ the same network architecture to extract features from RGB and point cloud data. Different from DenseFusion[11], we employ a weighted feature fusion approach for object pose regression. In the first learning stage, we utilize the Iterative Annotation Resolver from Weak6D [34] for training without 6D annotations. In the second stage, we introduce a self-supervised loss function based on a differentiable renderer for training the refinement network. The employment of the renderer can improve the training performance by providing superior feedback information, even without 6D annotations. The details of our method are described below.

B. Pose Estimation

The pose estimation model is illustrated in Fig. 2. RGB images provide rich color information but lack object geometry details. On the other hand, 3D point clouds directly represent object geometry but lack color information. Hence, effectively fusing these two complementary features in RGB-D input presents a significant challenge for the network. To address this challenge, We make improvements to the network structure of DenseFusion [11] and propose a novel pixel-wise weighted dense fusion architecture. This improved network is advantageous in scenarios involving occlusion or segmentation errors, where certain pixels may not contain object pose information. The pixel-wise weighted dense fusion efficiently leverages information implicitly present in the visible area, leading to improved accuracy and robustness in pose estimation for practical applications

Pose Estimation Network: We first use the mask generated by semantic segmentation to obtain the corresponding region of the target object in the RGB-D image. Then, we convert

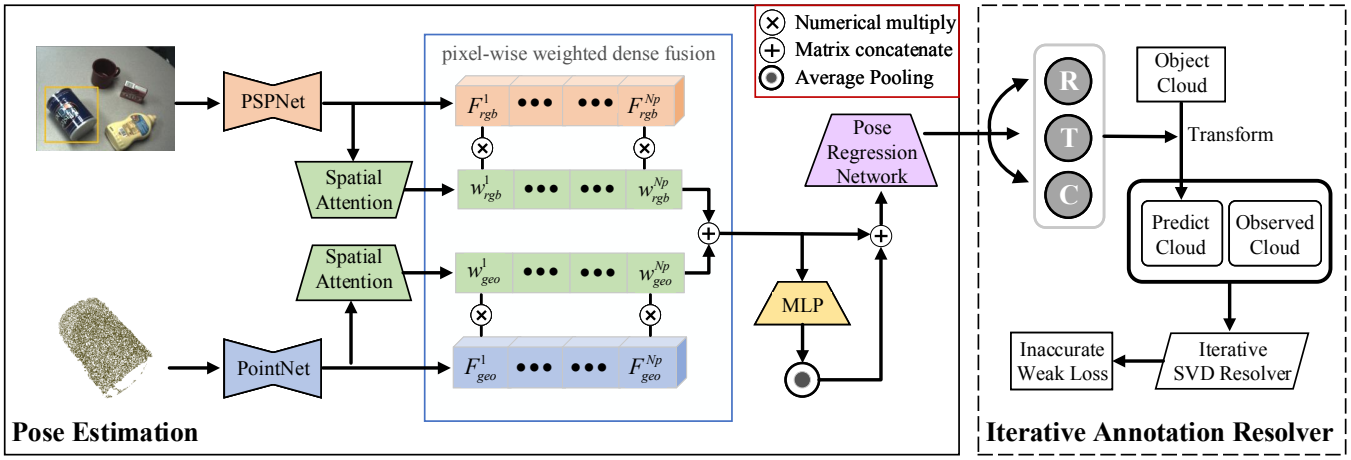

Fig. 2. Overview of the pose estimation method. Our model independently extracts features from RGB images and 3D point clouds. It then performs pixel-wise weighted dense fusion, with the pixel-wise weights generated using a spatial attention module. To optimize network parameters, we employ the Inaccurate Weak Loss generated by the Iterative Annotation Resolver.

the depth map into a 3D point cloud using the camera's intrinsic parameters. To optimize computational resources, we randomly select Np pixels alongside their corresponding 3D points. For feature extraction, we employ PSPNet [37] on RGB image clip and PointNet [38] on re-projected point cloud to extract color and geometric features.

Different from DenseFusion, in the pixel-wise dense fusion process of two features, we employ a spatial attention mechanism [39] to generate pixel-wise weights, which helps balance their contributions to the final pose. The shape of color feature $\mathbf{F}^{rgb}$ and geometric $\mathbf{F}^{geo}$ feature is $(Bs \times Cn \times Np)$, where Bs denotes the batch size, Cn denotes the number of channels, and Np denotes the number of sampled pixels. The spatial attention module performs both maximum pooling and average pooling for $\mathbf{F}^{rgb}$ and $\mathbf{F}^{geo}$ along the channel dimension, subsequently concatenating the pooled vectors. It is then followed by a convolutional operation and a sigmoid activation, ultimately producing the pixel-wise weight $\mathbf{W}^{rgb}$ and $\mathbf{W}^{geo}$ $(Bs \times Cn \times Np)$. For the j^{th} pixel-wise color feature $\mathbf{F}_j^{rgb}$ and pixel-wise geometric feature $\mathbf{F}_j^{geo}$, fusion calculation is expressed by the following formula:

$$\mathbf{F}_j^{fused} = (\mathbf{W}_j^{rgb} \cdot \mathbf{F}_j^{rgb}) \oplus (\mathbf{W}_j^{geo} \cdot \mathbf{F}_j^{geo}), \tag{1}$$

where $\oplus$ denotes tensor concatenation. The pixel-wise weighted dense fusion occurs on both color and geometric feature branches, and we further concatenate these two pixel-wise weighted dense features to regress the 6D object pose.

The fused pixel-wise features represent local features. Building upon this, we utilize a two-layer shared MLP and average pooling to extract global features. By combining the local and global features, we enhance the model's robustness in handling occlusion scenarios. Finally, three branches are employed to regress pixel-wise rotation **R**, transform **T**, and confidence **C** based on the fused features.

Inaccurate Weak Loss: In conventional methods, the loss function of the training network is typically constructed based on the error between the estimated pose and the ground truth pose. However, obtaining precise pose annotations can be costly. We address this issue by employing the Iterative Annotation Resolver proposed in Weak6D [34], which doesn't require any 6D annotations, to generate an Inaccurate Weak Loss. This approach enables network parameter optimization without relying on manual annotations.

The Iterative Annotation Resolver dynamically generates inaccurate learning objectives $\mathbf{P}_k^{weak} = [\mathbf{R}_k^{weak} | \mathbf{T}_k^{weak}]$ for the pixel-wise pose regression. For the prediction of the k^{th} pixel $\mathbf{P}_k = [\mathbf{R}_k | \mathbf{T}_k]$, the optimization objective is as follows:

$$L_k^p = \frac{1}{N} \sum_{j \in (0,N]} \left\| \mathbf{P}_k^{weak} \cdot \mathbf{X}_{k-j}^{mod} - \mathbf{P}_k \cdot \mathbf{X}_{k-j}^{mod} \right\|_2 \tag{3}$$

where $j \in (0, N]$. $\mathbf{X}_{k-j}^{mod}$ denotes the j^{th} point of the model, which can improve the robustness on occlusion.

To self-supervise the training of the confidence branch, we utilize the pixel-wise confidence output from the network to weigh the distance and add a regularization term. The final Loss function is as follows:

$$L = \frac{1}{N} \sum_{k \in (0,Np]} (L_{i-k}^p \cdot c_{i-k} - w \cdot \log(c_{i-k})) \tag{4}$$

C. Pose Refinement

Inspired by DenseFusion [11], we introduce a self-supervised pose refinement method based on the pixel-wise weighted architecture. Different from DenseFusion, our approach does not require pose annotations for training. Although the Iterative Annotation Resolver cannot generate high-precision pose annotations, it demonstrates an extremely fast running speed. To leverage its efficiency, we initially train the refinement network using the Iterative Annotation Resolver to accelerate convergence. Subsequently, we employ a self-supervised loss function based on a differentiable renderer to train the refinement network.

Iterative Refinement Network: We reverse transform the observed input point cloud using the output pose from the

estimation network. The color features from the estimation network and the transformed input point cloud are used as inputs to the refinement network. Similar to the estimation network, we employ the pixel-wise weighted dense fusion architecture to fuse the color features and geometry features. Since the estimation network has already trained the confidence branch, the refinement network only has two regression branches for R and T.

The output of the network is a transformation matrix that transforms the object from the object coordinate system to the camera coordinate system. This transformation is used to perform a reverse transformation on the input local point cloud. In an ideal scenario, the transformed local point cloud should coincide with the model point cloud in the object coordinate system. However, due to the imprecise nature of pose estimation, we can treat the transformed point cloud as another local point cloud under the camera coordinate system and feed it back to the network to predict a new pose. This iterative process can be repeated. As each reverse transformation of the input point cloud implicitly estimates the previous pose, the output pose of each iteration is a residual estimate based on the previously estimated pose. After K iterations, the final estimated pose is obtained as follows:

$$[R_K|\mathrm{T}_K] = [R_0|\mathrm{T}_0] \cdot \Delta[R_1|\mathrm{T}_1] \cdots \Delta[R_K|\mathrm{T}_K] \quad (5)$$

Once the loss value of the estimation network decreases to a certain extent, the refinement network can be jointly trained with the estimation network.

Self-Supervised Joint Loss: The primary challenge of training a network without pose annotation is to devise robust constraints for the network optimization process using input data. As illustrated in Fig. 3, we first transform the model using the network's output pose and employing a differentiable renderer to generate a rendered image. Then our objective is to design a self-supervised Loss function that aligns the rendered image with the masked camera image after semantic segmentation and aligns the input point cloud with the transformed model cloud.

Color Loss: We initially utilized the method employed in Self6D [31], which transforms both the rendered image and camera image from RGB color space to LAB color space, while ignoring the L channel. Additionally, we devised a bidirectional small neighborhood nearest-pixel error calculation method to further enhance the robustness. The calculation formula is as follows:

$$\begin{aligned} {}^{cam}_{ren}L_{(i,j)} &= \min_{m_1,m_2} ||\mathbf{I}^{cam}_{(i,j)} - \mathbf{I}^{ren}_{(i+m_1,j+m_2)}|| \\ {}^{ren}_{cam}L_{(i,j)} &= \min_{m_1,m_2} ||\mathbf{I}^{ren}_{(i,j)} - \mathbf{I}^{cam}_{(i+m_1,j+m_2)}|| \end{aligned}, \quad (6)$$

where $i \in (0, H]$, $j \in (0, W]$, $m_1, m_2 \in [0, M]$.For each pixel value $\mathbf{I}^{cam}_{(i,j)}$ located at (i, j) in the camera image $\mathbf{I}^{cam}$, we search for the value in a small neighborhood $(i - M \sim i + M, j - M \sim j + M)$ around the corresponding pixel in the rendered image $\mathbf{I}^{ren}$ and output the smallest error with it. Similarly, find the nearest value in the neighboring pixels

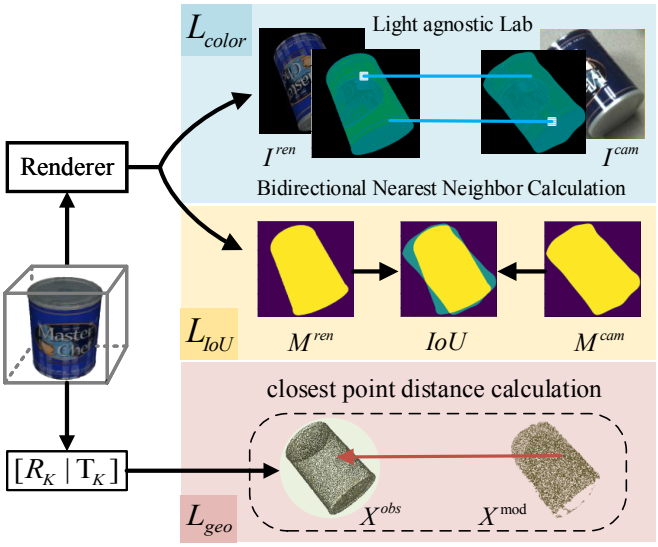

Fig. 3. Self-Supervised learning of refinement network. We transform the model using the network's output pose and employing a differentiable renderer to generate a rendered image and rendered mask. Then we devise a self-supervised color loss, IoU loss, and geometric loss to train the refinement network.

corresponding to the camera image $\mathbf{I}^{cam}$. Finally, the formula for calculating color loss is as follows:

$$L_{color} = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} ({}^{ren}_{cam}L_{(i,j)} + {}^{cam}_{ren}L_{(i,j)}) \quad (7)$$

IoU Loss: The image mask contains essential object pose information without any interference, providing strong constraints. To design the loss function, we compute the Intersection over Union (IoU) between the rendered image mask $\mathbf{M}^{ren}$ and the camera image mask $\mathbf{M}^{cam}$. The mask assigns a pixel value 1 to the object area and a pixel value 0 to the background area. The calculation formula is as follows:

$$L_{IOU} = 1 - \frac{\sum_{i=1}^{H} \sum_{j=1}^{W} \mathbf{M}^{ren}_{(i,j)} \cdot \mathbf{M}^{cam}_{(i,j)}}{\sum_{i=1}^{H} \sum_{j=1}^{W} (\mathbf{M}^{ren}_{(i,j)} + \mathbf{M}^{cam}_{(i,j)} - \mathbf{M}^{ren}_{(i,j)} \cdot \mathbf{M}^{cam}_{(i,j)})} \quad (8)$$

Geometry Loss: During the iteration process, we transform the model point cloud using the latest pose. To align the input local point cloud with the transformed global model point cloud, we calculate the mean distance from each point of the input point cloud to its closest neighbor on the transformed model point cloud as the loss function. For the output pose of the k^{th} iteration, the geometric loss calculation formula is as follows:

$$L_{geo} = \frac{1}{N} \sum_{j} \min_{h} ||\mathbf{R}_k \cdot \mathbf{X}^{mod}_j + \mathbf{T}_k - \mathbf{X}^{obs}_h||^2_2 \quad (10)$$

The total Loss function, which is a weighted sum of the three aforementioned losses with balance factors, is used to optimize network parameters. The total Loss function is as follows:

$$L = \lambda_1 \cdot L_{color} + \lambda_2 \cdot L_{IoU} + \lambda_3 \cdot L_{geo} \quad (11)$$

TABLE I
EVALUTION RESULT OF 6D POSE ON YCB-VIDEO DATASET

Object	with annotated real data		with annotated synthetic data		w/o annotation		
	DenseFuion[11]	DF(refine)[11]	Semi-Supervised[33]	Self6d++[32]	PureICP[14]	Ours	Ours(refine)
002_master_chef_can	93.6	**94.2**	**95.4**	88.8	73.1	85.9	**88.5**
003_cracker_box	91.9	**94.3**	88.6	**94.2**	61.9	75.9	**78.2**
004_sugar_box	95	**95.7**	95.3	**95.8**	81.5	80.8	**88.3**
005_tomato_soup_can	93.6	**93.9**	**93.6**	90.8	83.6	89.3	**91.9**
006_mustard_bottle	92.5	**93.6**	96.8	**98.6**	82.8	88.7	**91.8**
007_tuna_fish_can	94.1	**94.9**	96.2	**97.5**	90.8	86.9	**92.4**
008_pudding_box	93.8	**93.8**	89.7	**98.4**	**89.1**	87.9	88.2
009_gelatin_box	96.2	**97.2**	**96**	94	**90.5**	84.1	88.1
010_potted_meat_can	92.4	**93.3**	**90.1**	89.3	83	85.1	**90.1**
011_banana	85.4	**87.7**	93.2	**98.5**	**94.1**	81.2	85.3
019_pitcher_base	84.8	**85.3**	96.6	**98.9**	63.7	74.4	**76.4**
021_bleach_cleanser	91.1	**92**	92	**93.5**	69.5	81.8	**88.3**
024_bowl	86.9	**88.9**	87.4	**89.1**	62.2	73.2	**80**
025_mug	93	**93.3**	**96.7**	94.1	74.4	88.5	**89.1**
035_power_drill	88	**88.5**	91.6	**95.2**	75.2	81	**81.3**
036_wood_block	87.9	**89.9**	**89.8**	78.3	53.9	77.4	**83.1**
037_scissors	86.5	**88.7**	**81.7**	69.2	**92**	80.9	81.5
040_large_marker	95.6	**97**	**97.3**	87.5	**95.1**	85.8	92.4
051_large_clamp	89.4	**90.6**	72	**79.2**	85.9	82.6	**87**
052_extra_large_clamp	86.6	**88.5**	65.7	**87.3**	84	82.3	**86.3**
061_foam_brick	91.2	**91.3**	93.4	**95.5**	**91.2**	88.3	87
MEAN	91.5	**92.4**	90.9	**91.1**	79.1	83.3	**87.1**

IV. EXPERIMENTS

In the experimental section, we aim to address the following aspects: (1) Evaluate the performance of our pose estimation and refinement methods on unannotated data; (2) Assess the effectiveness of our pixel-wise weighted dense fusion architecture; (3) Compare the training efficiency of our method with other existing methods.

A. datasets

We utilize the widely used YCB-Video benchmark dataset for our research. This dataset comprises 21 objects, comprising 130,000 real-scene frames and 80,000 synthetic RGB-D frames. The complex and diverse scenes in the YCB-Video dataset present challenges for object pose estimation. Moreover, the training and testing data are collected from various scenes, enabling a robust assessment of the algorithm's performance.

B. Metric

We employ the area under the ADD-S curve (AUC) proposed by PoseCNN as the accuracy metric, with a maximum threshold of 0.1m. The ADD-S computes the mean distance from each target model point transformed by ground truth pose P to its closest neighbor on the model transformed by the estimated pose $\widehat{P}$. The calculation formula of ADD-S is as follows:

$$ADD-S = \frac{1}{m}\sum_{x_j \in M} min_{x_i \in M} \left\| Px_i - \widehat{P}x_j \right\| \qquad (12)$$

C. Implementation Details

In the experiments, we employ the semantic segmentation mask from the widely used BOP dataset directly. For the Iterative Annotation Resolver, We set the number of iterations to 20, and the hyper-parameter w of the Inaccurate Weak Loss for estimation network training was set to 0.015. In the training of the refinement network, We set the number of iterations K to 2, and the balance factors of the Self-supervised Loss are set to λ_1=0.0002, λ_2=0.1, and λ_3=1. All experiments are optimized using the Adam optimizer. In the PureICP experiment, the number of ICP iterations is set to 50.

D. Experimental Results

1) Effect of Self-Supervised Learning: We train our estimation and refinement networks using unannotated real-scene data from the YCB-Video dataset. Tab. I shows the evaluation results of our method on all 21 objects, along with the evaluation results of other methods. Our method demonstrates performance that is closely comparable to that of the supervised method.

With optimization through the Iterative Annotation Resolver, the estimation network achieves an accuracy of 83.3% according to the AUC metric. After being trained by the Iterative Annotation Resolver and Self-supervised Loss, the refinement network achieves an accuracy of 87.1%, showcasing an improvement of 3.8% compared to the performance before refinement. Additionally, these results surpass the direct estimation result achieved by the ICP algorithm, reaching a level of accuracy that is essential for practical applications. This experiment effectively demonstrates our method's capability to learn 6D object pose estimation from unannotated data and attain improved estimation accuracy through iterative refinement.

2) Effect of Pixel-Wise Weighted Dense Fusion: We first conduct ablation experiments on the YCB-Video dataset, removing the pixel-wise weighted module. The removed network is the same as the DenseFusion network. Then we

Fig. 4. Qualitative experimental results conducted on the YCB-Video dataset. The results show that our Self-Supervised learning method is close to the supervised method DenseFusion.

conduct experiments for the two networks under uniform configurations. Tab. II presents the comparative results of the original DenseFusion network and the pixel-wise weighted dense fusion network. The results show that our network significantly enhances the effect of self-supervised learning. In addition, we train our network using annotated synthetic data and real-scene data for inference. The results show that our network supervised by synthetic data achieved the same good result as the DenseFusion supervised by real-scene data. We analyzed this phenomenon for some assumptions. The pixel-wise weighted dense fusion can tune the feature distribution of the origin feature, which will adapt to the variations in the surface characteristics of objects in different environments, and enhance the generalization performance of the method.

TABLE II
EFFECT OF PIXEL-WISE WEIGHTED DENSE FUSION

Object	w/o weighted	with weighted	DF + real GT	ours + syn GT
scissors	**81.2**	80.9	88.7	**89.6**
large_marker	77.6	**85.8**	**97**	96.4
large_clamp	75.6	**82.6**	**90.6**	89.5
extra_large_clamp	77.3	**82.3**	88.5	**89.2**
foam_brick	84	**88.3**	91.3	**91.8**
MEAN	79.1	**84**	91.2	**91.3**

3) Training Efficiency: We conduct a comparative assessment of training efficiency between Iterative Annotation Resolver and differentiable renderer. Both the Iterative Annotation Resolver and the differentiable renderer are developed based on PyTorch3D and subjected to training efficiency evaluations on an NVIDIA GeForce RTX 2080 Ti GPU. As shown in Tab. III, Training with a differentiable renderer takes 6 times longer than the Iterative Annotation Resolver and consumes 3 times of memory. Notably, in practical experimentation, under the acceleration of Iterative Annotation Resolver, we only need to train one epoch based on rendering in the whole training process of the estimation and refinement network, so the training efficiency of our method is far better than the self-supervised learning method based on rendering.

4) Qualitative Evaluation: Fig. 4 illustrates the inference results of our method and the DenseFusion method across various test samples. We utilize the estimated pose to transform the object model and back-project it onto the RGB image. The results indicate that our self-supervised learning approach closely approximates the results achieved by the supervised DenseFusion method. However, our method encounters challenges in accurately estimating the pose of symmetric objects within occluded scenes, which will be a problem we need to solve in the future.

TABLE III
TRAINING EFFICIENCY

	Time usage	Memory usage
Iterative Annotation Resolver	0.049ms	2061MB
Render	0.300ms	6385MB

V. CONCLUSIONS

This paper introduces a novel approach for 6D object pose estimation. Leveraging the pixel-wise weighted dense fusion architecture, this method encompasses the entire process of pose estimation and refinement, and it can be trained using unannotated RGB-D data through an Iterative Annotation Resolver and a Self-supervised Loss based on the differentiable renderer. We assessed the performance of our method on the YCB-Video dataset, and the experimental results demonstrated the practicality of our method in achieving meaningful outcomes even without annotations.

ACKNOWLEDGMENT

This work was supported by the National Key Research and Development Program of China under Grant 2018AAA0102504, in part by the National Natural Science Foundation of China (U1964203, 62203089, 62003073); in part by the Project funded by China Postdoctoral Science Foundation under Grant 2021M700695, and in part by the Sichuan Science and Technology Program (2022NSFSC0890, 2021YFS0383, 2023YFG0024, 2022YFS0570); in part by the Fundamental Research Funds for the Central Universities (ZYGX2022YGRH003, ZYGX2021YGLH003).

REFERENCES

[1] J. Yu, K. Weng, G. Liang and G. Xie, "A vision-based robotic grasping system using deep learning for 3D object recognition and pose estimation," in *ROBIO*, Shenzhen, China, 2013, pp. 1175-1180, DOI: 10.1109/ROBIO.2013.6739623.

[2] Y. Peng, X. Yang, S. Wei, X. Gao, W. Li and J. Z. Wen, "6D hybrid pose estimation in cluttered industrial scenes for robotic grasping," in *IARCE*, Chengdu, China, 2022, pp. 19-23, DOI: 10.1109/IARCE57187.2022.00014.

[3] Y. Liu, Y. Yixuan and M. Liu, "Ground-aware monocular 3D object detection for autonomous driving," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 919-926, April 2021, DOI: 10.1109/LRA.2021.3052442.

[4] V. Ravi Kumar et al., "Omnidet: surround view cameras based multi-task visual perception network for autonomous driving," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2830-2837, April 2021, DOI: 10.1109/LRA.2021.3062324.

[5] Y. Su, J. Rambach, N. Minaskan, P. Lesur, A. Pagani and D. Stricker, "Deep multi-state object pose estimation for augmented reality assembly," in *ISMAR-Adjunct*, Beijing, China, 2019, pp. 222-227, DOI: 10.1109/ISMAR-Adjunct.2019.00-42.

[6] Y. Lu, S. Kourian, C. Salvaggio, C. Xu and G. Lu, "Single image 3d vehicle pose estimation for augmented reality," in *GlobalSIP*, Ottawa, ON, Canada, 2019, pp. 1-5, DOI: 10.1109/GlobalSIP45357.2019.8969201, DOI: 10.1109/GlobalSIP45357.2019.8969201.

[7] D. G. Lowe, "Object recognition from local scale-invariant features," in *ICCV*, Kerkyra, Greece, 1999, pp. 1150-1157 vol.2, DOI: 10.1109/ICCV.1999.790410.

[8] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking," in *ICCV*, Volume 1, Beijing, China, 2005, pp. 1508-1515 Vol. 2, DOI: 10.1109/ICCV.2005.104.

[9] R. B. Rusu, N. Blodow and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *ICRA*, Kobe, Japan, 2009, pp. 3212-3217, DOI: 10.1109/ROBOT.2009.5152473.

[10] A. Aldoma et al., "CAD-model recognition and 6DoF pose estimation using 3D cues," in *ICCVW*, Barcelona, Spain, 2011, pp. 585-592, DOI: 10.1109/ICCVW.2011.6130296.

[11] C. Wang, D. Xu, Y. Zhu, M. Roberto, C. Lu, L. Fei-Fei and S. Savarese, "Densefusion: 6D object pose estimation by iterative dense fusion," in *CVPR*, Long Beach, CA, USA, 2019, pp. 3338-3347, DOI: 10.1109/CVPR.2019.00346.

[12] Y. He, H. Huang, H. Fan, Q. Chen and J. Sun, "FFB6D: a full flow bidirectional fusion network for 6D pose estimation," in *CVPR*, Nashville, TN, USA, 2021, pp. 3002-3012, DOI: 10.1109/CVPR46437.2021.00302.

[13] Y. Xiang, T. Schmidt, V. Narayanan and D. Fox, "PoseCNN: a convolutional neural network for 6D object pose estimation in cluttered scenes," in *RSS*, Pittsburgh, PA, USA, 2018, DOI: 10.15607/RSS.2018.XIV.019.

[14] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239-256, Feb. 1992, DOI: 10.1109/34.121791.

[15] J. Park, Q. -Y. Zhou and V. Koltun, "Colored point cloud registration revisited," in *ICCV*, Venice, Italy, 2017, pp. 143-152, DOI: 10.1109/ICCV.2017.25.

[16] Y. Li, G. Wang, X. Ji, Y. Xiang, D. Fox, "DeepIM: deep iterative matching for 6D pose estimation," *Int. J. Comput. Vis.*, vol. 128, pp. 657-678, Sep. 2019, DOI: 10.1007/s11263-019-01250-9.

[17] A. Trabelsi, M. Chaabane, N. Blanchard and R. Beveridge, "A pose proposal and refinement network for better 6D object pose estimation," in *WACV*, Waikoloa, HI, USA, 2021, pp. 2381-2390, DOI: 10.1109/WACV48630.2021.00243.

[18] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981, DOI: 10.1145/358669.358692

[19] Vincent Lepetit, F. Moreno-Noguer and P. Fua, "EPnP:an accurate o(n) solution to the pnp problem," *Int. J. Comput. Vis.*, vol. 81, pp. 155-166, 2009, DOI: 10.1007/s11263-008-0152-6.

[20] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: speeded up robust features," in *ECCV*, Graz, Austria, 2006, pp. 404–417, DOI: 10.1007/11744023_32

[21] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *ICCV*, Barcelona, Spain, 2011, pp. 2564-2571, DOI: 10.1109/ICCV.2011.6126544.

[22] S. Salti, F. Tombari, and L. Di Stefano, "Shot: unique signatures of histograms for surface and texture description," *Computer Vision and Image Understanding*, vol. 125, pp. 251–264, 2014, DOI: 10.1016/j.cviu.2014.04.011

[23] A. Zeng, S. Song, M. Nie?ner, M. Fisher, J. Xiao and T. Funkhouser, "3DMatch: learning local geometric descriptors from RGB-D reconstructions," in *CVPR*, Honolulu, HI, USA, 2017, pp. 199-208, DOI: 10.1109/CVPR.2017.29.

[24] Z. Li, G. Wang and X. Ji, "CDPN: coordinates-based disentangled pose network for real-time RGB-based 6-DoF object pose estimation," in *ICCV*, Seoul, Korea (South), 2019, pp. 7677-7686, DOI: 10.1109/ICCV.2019.00777.

[25] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song and L. J. Guibas, "Normalized object coordinate space for category-level 6D object pose and size estimation," in *CVPR*, Long Beach, CA, USA, 2019, pp. 2637-2646, DOI: 10.1109/CVPR.2019.00275.

[26] W. Chen, X. Jia, H. J. Chang, J. Duan and A. Leonardis, "G2L-Net: global to local network for real-time 6d pose estimation with embedding vector features," in *CVPR*, Seattle, WA, USA, 2020, pp. 4232-4241, DOI: 10.1109/CVPR42600.2020.00429.

[27] G. Gao, M. Lauri, Y. Wang, X. Hu, J. Zhang and S. Frintrop, "6D object pose regression via supervised learning on point clouds," in *ICRA*, Paris, France, 2020, pp. 3643-3649, DOI: 10.1109/ICRA40945.2020.9197461.

[28] C. Wang et al., "6-PACK: category-level 6D pose tracker with anchor-based keypoints," in *ICRA*, Paris, France, 2020, pp. 10059-10066, DOI: 10.1109/ICRA40945.2020.9196679.

[29] S. Peng, X. Zhou, Y. Liu, H. Lin, Q. Huang and H. Bao, "PVNet: pixel-wise voting network for 6DoF object pose estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 6, pp. 3212-3223, 1 June 2022, DOI: 10.1109/TPAMI.2020.3047388.

[30] Y. He, W. Sun, H. Huang, J. Liu, H. Fan and J. Sun, "PVN3D: a deep point-wise 3D keypoints voting network for 6DoF pose estimation," in *CVPR*, Seattle, WA, USA, 2020, pp. 11629-11638, DOI: 10.1109/CVPR42600.2020.01165.

[31] G. Wang, F. Manhardt, J. Shao, X. Ji, N. Navab and F. Tombari, "Self6D: self-supervised monocular 6D object pose estimation," in *ECCV*, Glasgow, UK, 2020, pp. 108–125, DOI: 10.1007/978-3-030-58452-8_7

[32] G. Wang, F. Manhardt, X. Liu, X. Ji and F. Tombari, "Occlusion-aware self-supervised monocular 6D object pose estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, DOI: 10.1109/TPAMI.2021.3136301.

[33] G. Zhou, D. Wang, Y. Yan, H. Chen and Q. Chen, "Semi-supervised 6D object pose estimation without using real annotations," *IEEE Trans Circuits Syst Video Technol*, vol. 32, no. 8, pp. 5163-5174, Aug. 2022, DOI: 10.1109/TCSVT.2021.3138129.

[34] F. Mu, R. Huang, K. Shi, X. Li, J. Qiu and H. Cheng, "Weak6D: weakly supervised 6D pose estimation with iterative annotation resolver," *IEEE Robot. Autom. Lett.*, vol. 8, no. 3, pp. 1463-1470, March 2023, DOI: 10.1109/LRA.2022.3190094.

[35] F. Wang, X. Zhang, T. Chen, Z. Shen, S. Liu, and Z. He, "Kvnet: an iterative 3d keypoints voting network for real-time 6-DoF object pose estimation," *Neurocomputing*, vol. 530, pp. 11–22, 2023, DOI: 10.1016/j.neucom.2023.01.036

[36] L. Lipson, Z. Teed, A. Goyal and J. Deng, "Coupled iterative refinement for 6D multi-object pose estimation," in *CVPR*, New Orleans, LA, USA, 2022, pp. 6718-6727, DOI: 10.1109/CVPR52688.2022.00661.

[37] H. Zhao, J. Shi, X. Qi, X. Wang and J. Jia, "Pyramid scene parsing network," in *CVPR*, Honolulu, HI, USA, 2017, pp. 6230-6239, DOI: 10.1109/CVPR.2017.660.

[38] R. Q. Charles, H. Su, M. Kaichun and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," in *CVPR*, Honolulu, HI, USA, 2017, pp. 77-85, DOI: 10.1109/CVPR.2017.16.

[39] S. Woo, J. Park, J. Lee and I. S. Kweon, "CBAM: convolutional block attention module," in *ECCV*, Munich, Germany, 2018, pp. 3-19, DOI: 10.1007/978-3-030-01234-2_1