SMOC-Net: Leveraging Camera Pose for Self-Supervised Monocular Object Pose Estimation

Tao Tan[1,2], Qiulei Dong[1,2,3]
[1]School of Artificial Intelligence, UCAS
[2]State Key Laboratory of Multimodal Artificial Intelligence Systems, CASIA
[3]Center for Excellence in Brain Science and Intelligence Technology, CAS
tantao2022@ia.ac.cn,qldong@nlpr.ia.ac.cn,Corresponding author:Qiulei Dong

Abstract

Recently, self-supervised 6D object pose estimation, where synthetic images with object poses (sometimes jointly with un-annotated real images) are used for training, has attracted much attention in computer vision. Some typical works in literature employ a time-consuming differentiable renderer for object pose prediction at the training stage, so that (i) their performances on real images are generally limited due to the gap between their rendered images and real images and (ii) their training process is computationally expensive. To address the two problems, we propose a novel Network for Self-supervised Monocular Object pose estimation by utilizing the predicted Camera poses from un-annotated real images, called SMOC-Net. The proposed network is explored under a knowledge distillation framework, consisting of a teacher model and a student model. The teacher model contains a backbone estimation module for initial object pose estimation, and an object pose refiner for refining the initial object poses using a geometric constraint (called relative-pose constraint) derived from relative camera poses. The student model gains knowledge for object pose estimation from the teacher model by imposing the relative-pose constraint. Thanks to the relative-pose constraint, SMOC-Net could not only narrow the domain gap between synthetic and real data but also reduce the training cost. Experimental results on two public datasets demonstrate that SMOC-Net outperforms several state-of-the-art methods by a large margin while requiring much less training time than the differentiable-renderer-based methods.

1. Introduction

Monocular 6D object pose estimation is a challenging task in the computer vision field, which aims to estimate object poses from single images. According to whether real images with ground-truth object poses are given for model training, the existing works for monocular object pose estimation in literature could be divided into two categories: fully-supervised methods [17, 35] which are trained by utilizing annotated real images with ground-truth object poses, and self-supervised methods [16, 34] which are trained by utilizing synthetic images with object poses (sometimes jointly with un-annotated real images). Since it is very time-consuming to obtain high-quality object poses as ground truth, self-supervised monocular object pose estimation has attracted increasing attention recently [16, 33, 39].

Some existing methods for self-supervised monocular object pose estimation [16, 31] use only synthetic images with object poses (which are generated via Blender [22] or some other rendering tools [24, 29]) for training. However, due to the domain gap between real and synthetic data, the performances of these self-supervised methods are significantly lower compared to the fully-supervised methods [22, 35]. Addressing this domain gap problem, a few recent self-supervised methods [33, 34, 39] jointly use synthetic images with object pose and un-annotated real images at their training stage, where a differentiable renderer [19] is introduced to provide a constraint on the difference between real and rendered images. Although these methods could alleviate the domain gap problem by utilizing the introduced differentiable renderer, they still have to be confronted with the following two problems: (i) There still exists a noticeable gap between real images and rendered images by the differentiable renderer, so that their performances on object pose estimation are still limited; (ii) Much time has to be spent on differentiable rendering during training, so that the training costs of these methods are quite heavy.

To address the above problems, this paper proposes a novel Network for Self-supervised Monocular Object pose estimation by utilizing the predicted Camera poses from un-annotated real images, called SMOC-Net. The SMOC-Net is designed via the knowledge distillation technique, con-

sisting of a teacher model and a student model. Under the teacher model, a backbone pose estimation module is introduced to provide an initial estimation on the pose of the image object. Then, a geometric constraint (called relative-pose constraint) on object poses is mathematically derived from relative camera poses which are calculated by a typical structure from motion method (here, we straightforwardly use COLMAP [25, 26]), and a camera-pose-guided refiner is further explored to refine the initial object pose based on this constraint. The student model simply employs the same architecture as the backbone estimation module of the teacher model, and it learns knowledge from the teacher model by imposing the relative-pose constraint, so that it could estimate object poses as accurately and fast as possible. Once the proposed SMOC-Net has been trained, only its student model is used to predict the object pose from an arbitrary testing image.

In sum, the main contributions in this paper include:

1. We design the relative-pose constraint on object poses under the knowledge distillation framework for self-supervised object pose estimation. And it could narrow the domain gap between synthetic and real data to some extent.

2. According to the designed relative-pose constraint, we explore the camera-pose-guided refiner, which is able to refine low-accuracy object poses effectively.

3. By jointly utilizing the camera-pose-guided refiner and the above object pose constraint, we propose the SMOC-Net for monocular 6D object pose estimation. Experimental results in Sec. 4 demonstrate that the proposed SMOC-Net does not only outperform several state-of-the-art methods when only synthetic images with object poses and un-annotated real images are used for training, but also perform better than three state-of-the-art fully-supervised methods on the public dataset LineMOD [8].

2. Related Work

In this section, we review the existing fully-supervised and self-supervised methods for monocular 6D object pose estimation in literature, respectively.

2.1. Fully-Supervised 6D Object Pose Estimation

Fully-supervised object pose estimation methods use real RGB(D) images with object poses for training. Some early works directly regressed object pose with various Convolutional Neural Networks (CNNs) such as PoseCNN [37] and SSD-6D [13]. Xiang *et al.* [37] proposed the PoseCNN for 6D object pose estimation, which estimated the 3D translation by predicting the distance between object center in the image and the camera, and the 3D rotation was estimated by regressing a quaternion representation. Kehl *et al.* [13] proposed the SSD-6D, where rotation estimation was treated as a classification problem by discretizing the rotation space into classifiable viewpoint bins.

For improving pose estimation accuracy further, many methods first established 2D-3D correspondences using CNNs, and then solved the object pose via Perspective-n-Point (PnP) algorithms. Keypoint-based methods [22, 28, 32] established correspondences by detecting keypoints in 2D images. Peng *et al.* [22] proposed a pixel level voting network (PVNet) by using the direction vector field to predict keypoints, which achieved good performance under severe truncation and occlusion. Song *et al.* [28] proposed the HybridPose by using hybrid representation such as keypoints, edge vectors, and symmetry correspondences. Some other methods predicted the corresponding 3D coordinates of each pixel in 2D images [17, 21, 40] or dense UV maps [41] to establish dense correspondences. Li *et al.* [17] proposed CDPN, which treated rotation and translation estimations as two different tasks. Rotation was calculated from dense correspondences, while translation was directly regressed from the target region. Hodan *et al.* [11] proposed EPOS, which represented the target object by compact surface fragments for better handling symmetries. To achieve end-to-end training, a Patch-PnP module consisting of CNNs and fully connected layers was proposed in [35]. Chen et al. [4] proposed an Epro-PnP module whose output was the probability density distribution of object pose and also achieved end-to-end training. Su *et al.* [30] proposed a discrete descriptor to represent the object surface densely, which was used to establish accurate and robust correspondences. Huang et al. [18] proposed to predict 3D object coordinates at 3D query points sampled in the camera frustum, which could efficiently handle occlusions. Shun et al. [12] utilized a deep texture rendering and differentiable Levenberg-Marquardt optimization to refine the object poses fast and accurately. Xu et al. [38] proposed a framework based on a recurrent neural network for object pose refinement, which was robust to erroneous initial poses and occlusions.

2.2. Self-Supervised 6D Object Pose Estimation

To avoid the data annotation problem in fully-supervised object pose estimation, many self-supervised methods for object pose estimation have been proposed. Some self-supervised methods [16, 23, 31] only used synthetic images with object poses as training data. Sundermeyer *et al.* [31] proposed the AAE method by utilizing an auto-encoder, where only synthetic data was used for training. It employed a series of domain adaption techniques to reduce the domain gap between real and synthetic images. Rambach *et al.* [23] proposed to estimate object pose from synthetic single channel images, showing that proper representation could also reduce the domain gap. Li *et al.* [16] proposed SD-Pose, which preprocessed input images by multi-level semantic representations and achieved better performance.

To further narrow the domain gap, a few recent self-

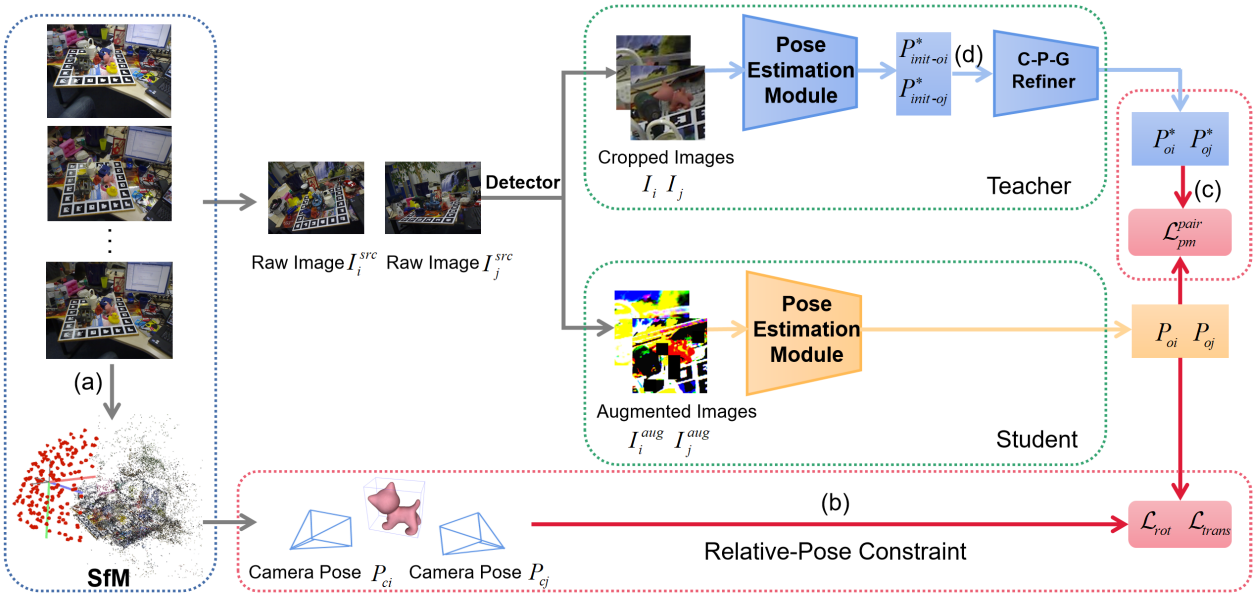

Figure 1. Architecture of the proposed SMOC-Net. It consists of a teacher model and a student model. The teacher model consists of a backbone pose estimation module and a camera-pose-guided (C-P-G) refiner. The student model simply employs the same architecture as the backbone estimation module of the teacher model. For a set of multi-view real images $\{I_1^{src}, ..., I_n^{src}\}$, their corresponding camera poses $\{P_{c1}, ..., P_{cn}\}$ are calculated by an SfM method (a). Then, a 2D object detector (Yolov4 [1]) is used to extract object regions from the input images, and a set of object patch pairs are obtained for self-supervised training. The predicted object poses $\{P_{oi}, P_{oj}\}$ are self-supervised with camera poses $\{P_{ci}, P_{cj}\}$ (b) and pseudo pose labels $\{P^*_{oi}, P^*_{oj}\}$ (c) that have been refined by the camera-pose-guided refiner (d).

supervised methods [33, 34, 39] jointly used synthetic images with object poses and un-annotated real images at their training stage. In [34], the network was first trained on synthetic data in a fully-supervised manner and then self-supervised with un-annotated real data by seeking a visually and geometrically optimal alignment between real images and rendered images by a differentiable renderer. But depth images were required during self-supervised training. Sock *et al.* [27] proposed a two-stage self-supervised method, which also used a differentiable renderer to establish pose consistency between rendered and real images. Beyond using image-level consistency for self-supervised pose estimation, Yang *et al.* [39] proposed a self-supervised network DSC-PoseNet based on dual-scale keypoint consistency. But it suffered dramatic performance degradation when there was occlusion. Wang *et al.* [33] revised [34] and proposed Self6D++, which used the pose estimation network GDR-Net [35] and noisy student training to improve accuracy and robustness. However, there still exists a gap between real images and rendered images by the differentiable renderer. And additionally, differentiable rendering is a computationally expensive process. Addressing these issues, this paper proposes a novel self-supervised object pose estimation by imposing a derived geometric constraint from camera poses, which would be described in detail in the following section.

3. Methodology

In this section, we propose the SMOC-Net for self-supervised monocular object pose estimation. Firstly, the architecture of the proposed SMOC-Net is introduced. Then, we present the derived geometric constraint from camera poses and the camera-pose-guided refiner respectively. Finally, the total loss function is described.

3.1. Architecture

The SMOC-Net is explored under a knowledge distillation framework, which contains a teacher model and a student model, and its architecture is shown in Fig. 1. As seen from this figure, an object detector is firstly used for extracting the object regions in the input images, and we simply use Yolo-v4 [1] here as the object detector as done in some existing object pose estimation works [33, 35]. The teacher model consists of a backbone estimation module and a camera-pose-guided refiner. The backbone estimation module is used to calculate an initial object pose estimation, and we simply use the existing method GDR-Net [35] as the backbone estimation module here. The camera-pose-guided refiner is explored to refine the initial object pose obtained from the backbone estimation module, by utilizing a derived geometric constraint from relative camera poses (called relative-pose constraint). The student model

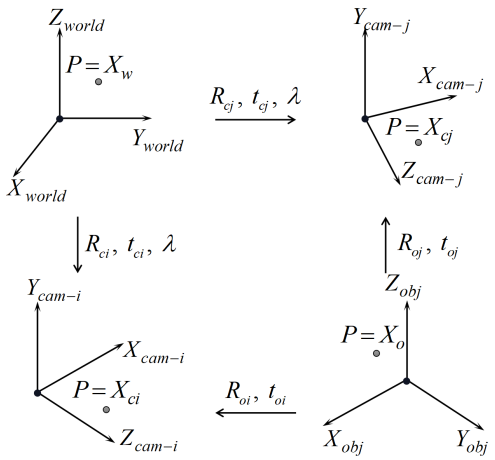

Figure 2. World, object and camera coordinate systems. For a 3D point P, its coordinates could be transformed from object/world coordinate systems to camera coordinate systems by corresponding absolute object/camera poses in the camera/world systems.

employs the same architecture as the backbone estimation module, and it gains knowledge from the teacher model by utilizing the same relative-pose constraint.

At the training stage, the proposed SMOC-Net is trained by utilizing both synthetic data [10] with labels and a set of un-annotated real multi-view images. Firstly, the backbone GDR-Net and the detector Yolo-v4 are trained singly by only utilizing the synthetic RGB data as done in [33,34]. Then, the classic SfM (Structure from Motion) method COLMAP [5,25,26] is utilized to calculate the corresponding absolute camera poses from the training set of multi-view real images, and the object detector is implemented for extracting object patches from the input real images. According to the calculated camera poses as well as the corresponding original real object patches, we could straightforwardly obtain a set of object patch pairs with pseudo relative camera poses (which could be calculated according to the obtained absolute camera poses). Next, unlike the existing works [33, 34, 39] that deal with each training image singly, the proposed method deals with pairs of training patches. Finally, the student model is trained with the set of augmented object patch pairs with pseudo relative poses for learning knowledge from the teacher model.

At the testing stage, only the student module is used for predicting object pose from each monocular RGB image. In Sections 3.2 and 3.3, both the relative-pose constraint and the camera-pose-guided refiner would be introduced, respectively, in detail.

3.2. Relative-Pose Constraint

Given a set of N object patch pairs with the corresponding absolute camera poses in the world coordinate system, we could obtain a geometric constraint on the object relative pose (called relative-pose constraint) for each patch pair as follows:

For an arbitrary pair of object patches, without loss of generality, let i and j $(i < j; i = 1, ..., N-1; j = i+1, ..., N)$ denote the indices of the two patches. Then as shown in Fig. 2, for an arbitrary 3D object point, let X_{ci} and X_{cj} denote its coordinates in the two camera coordinate systems corresponding to the two input patches, and let X_o and X_w denote its coordinates in the object and world coordinate systems respectively.

Let $[R_{oi}, t_{oi}]$ and $[R_{oj}, t_{oj}]$ represent the absolute object poses in the two camera systems. Then, we have

$$X_{ci} = R_{oi}X_o + t_{oi}, \tag{1}$$
$$X_{cj} = R_{oj}X_o + t_{oj}. \tag{2}$$

Let $[R_{ci}, t_{ci}]$ and $[R_{cj}, t_{cj}]$ represent the absolute camera poses in the world system, which are calculated by the SfM method COLMAP [5, 25]. It has to be pointed out that the calculated translations by COLMAP are not equal to the ground-truth translation, but up to a scale. Hence, we have

$$X_{ci} = R_{ci}X_w + \lambda t_{ci}, \tag{3}$$
$$X_{cj} = R_{cj}X_w + \lambda t_{cj}, \tag{4}$$

where λ is a scale factor. According to Eqn. (1) and Eqn. (3), we have

$$X_w = R_{ci}^{-1}R_{oi}X_o + R_{ci}^{-1}(t_{oi} - \lambda t_{ci}). \tag{5}$$

According to Eqn. (2) and Eqn. (4), we have

$$X_w = R_{cj}^{-1}R_{oj}X_o + R_{cj}^{-1}(t_{oj} - \lambda t_{cj}). \tag{6}$$

According to Eqn. (5) and Eqn. (6), we have

$$R_{ci}^{-1}R_{oi} = R_{cj}^{-1}R_{oj}, \tag{7}$$
$$R_{ci}^{-1}(t_{oi} - \lambda t_{ci}) = R_{cj}^{-1}(t_{oj} - \lambda t_{cj}). \tag{8}$$

Eqn. (7) could be rewritten as

$$R_{oi}R_{oj}^{-1} = R_{ci}R_{cj}^{-1}. \tag{9}$$

Eqn. (8) could be rewritten as

$$R_{ci}^{-1}t_{oi} - R_{cj}^{-1}t_{oj} = \lambda(R_{ci}^{-1}t_{ci} - R_{cj}^{-1}t_{cj}). \tag{10}$$

To remove the scale factor λ, both sides of the Eqn. (10) are normalized as

$$\frac{R_{ci}^{-1}t_{oi} - R_{cj}^{-1}t_{oj}}{||R_{ci}^{-1}t_{oi} - R_{cj}^{-1}t_{oj}||_2} = \frac{R_{ci}^{-1}t_{ci} - R_{cj}^{-1}t_{cj}}{||R_{ci}^{-1}t_{ci} - R_{cj}^{-1}t_{cj}||_2}, \tag{11}$$

where $||\cdot||_2$ is the L_2 norm. It is noted that once the camera poses $[R_{ci}, t_{ci}]$ and $[R_{cj}, t_{cj}]$ are known, Eqn. (9) and Eqn. (11) provide constraints on relative object rotation and translation respectively. Hence, they are used together as our relative-pose constraint.

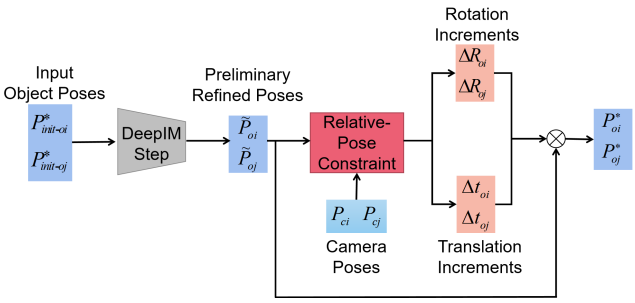

Figure 3. Architecture of the explored camera-pose-guided refiner. For a pair of 6D object poses $\{P^*_{init\text{-}oi}, P^*_{init\text{-}oj}\}$, they are first refined by DeepIM [15]. The refined poses $\{\widetilde{P}_{oi}, \widetilde{P}_{oj}\}$ are then further updated using rotation increments and translation increments calculated according to the relative-pose constraint.

3.3. Camera-Pose-Guided Refiner

In this section, we propose an iterative object pose refiner (called camera-pose-guided refiner), and its architecture is shown in Fig. 3. Given a set of N object patch pairs with the corresponding absolute camera poses in the world coordinate system, and their corresponding initial object poses are estimated by the teacher backbone estimation module. We could further refine the initial poses of each patch pair by utilizing the above relative-pose constraint as follows:

At each iteration, the object poses are firstly refined by DeepIM [15], which is pre-trained on synthetic RGB data. Let $\widetilde{P}_{oi} = [\widetilde{R}_{oi}, \widetilde{t}_{oi}]$ and $\widetilde{P}_{oj} = [\widetilde{R}_{oj}, \widetilde{t}_{oj}]$ represent the refined absolute object poses of two input patches. Then, we further refine the absolute object rotations and translations of the two patches with the corresponding absolute camera poses, respectively.

Rotation Refinement Let $R_{ij} = R_{ci}R_{cj}^{-1}$ denotes the relative camera rotation of two camera systems. In the rotation refinement, we seek to apply slight increments to the object rotations $\{\widetilde{R}_{oi}, \widetilde{R}_{oj}\}$. At each iteration, let $\{\Delta R_{oi}, \Delta R_{oj}\}$ represent the referred slight increments. Then, the object rotations would be updated as $\{\widetilde{R}_{oi}\Delta R_{oi}, \widetilde{R}_{oj}\Delta R_{oj}\}$. According to the relative rotation constraint in Eqn. (9), we have the following constraint equation:

$$\Delta R_{oi}^{-1}\Delta R_{ij}\Delta R_{oj} = E, \tag{12}$$

where $\Delta R_{ij} = \widetilde{R}_{oi}^{-1}R_{ij}\widetilde{R}_{oj}$. This is an under-constrained equation whose solution is not unique, hence, an auxiliary condition $\Delta R_{oi}^{-1} = \Delta R_{oj}$ is introduced for guaranteeing a unique solution, considering that both the increments ΔR_{oi} and ΔR_{oj} at each iteration change slightly. Then, considering that Eqn. (12) is a nonlinear equation which is hard to solve, we compute the approximate solution by solving the following minimization problem:

$$\min_{\Delta R_{oi}, \Delta R_{oj}} ||\omega(\Delta R_{oi}^{-1}\Delta R_{ij}\Delta R_{oj})||_2^2, \tag{13}$$

where $\omega(R)$ is the axis-angle representation of 3D rotation R. Let $\{\Delta\omega_{oi}, \Delta\omega_{oj}\}$ denote the axis-angle representation of $\{\Delta R_{oi}, \Delta R_{oj}\}$, and $\Delta\omega_{ij}$ denote the axis-angle representation of ΔR_{ij}. The concatenation of $\Delta\omega_{oi}$ and $\Delta\omega_{oj}$ into a 6×1 vector is denoted as $\Delta\Omega$. Therefore, our problem is to solve the following minimization problem:

$$\min_{\Delta\Omega} ||r_{ij}(\Delta\Omega)||_2^2, \tag{14}$$

where

$$r_{ij}(\Delta\Omega) = \omega(R(-\Delta\omega_{oi})R(\Delta\omega_{ij})R(\Delta\omega_{oj})). \tag{15}$$

Inspired by Avishek et al. [3], the above minimization problem is transformed to solve the following problem using quasi-Newton method:

$$\min_{\Delta\Omega} ||r_{ij}(\mathbf{0}) + \mathcal{J}r_{ij}(\mathbf{0})^T\Delta\Omega||_2^2. \tag{16}$$

It is equivalent to solve a linear system of equations:

$$Jr_{ij}(\mathbf{0})^T\Delta\Omega = -r_{ij}(\mathbf{0}), \tag{17}$$

where $Jr_{ij}(\mathbf{0})$ is the Jacobian of $r_{ij}(\Delta\Omega)$ at the current point $\Delta\Omega = \mathbf{0}$, and

$$\begin{aligned} Jr_{ij}(\mathbf{0})^T &\approx \frac{||\Delta\omega_{ij}||_2}{2}\cot(\frac{||\Delta\omega_{ij}||_2}{2})[I\ -I] \\ &\approx [I\ -I]. \end{aligned} \tag{18}$$

Therefore, the final approximate solution is

$$\Delta\omega_{oi} = -\Delta\omega_{oj} = -\frac{\Delta\omega_{ij}}{2}, \tag{19}$$

$$\Delta R_{oi} = \Delta R_{oj}^{-1} = R(-\frac{\Delta\omega_{ij}}{2}). \tag{20}$$

Translation Refinement In the translation refinement, we seek to apply slight increments to object translations $\{\widetilde{t}_{oi}, \widetilde{t}_{oj}\}$. At each iteration, let $\{\Delta t_{oi}, \Delta t_{oj}\}$ represent the referred slight increments. Then, the object translations would be updated as $\{\widetilde{t}_{oi} + \Delta t_{oi}, \widetilde{t}_{oj} + \Delta t_{oj}\}$. According to the relative translation constraint in Eqn. (11), we have the following constraint equation:

$$\frac{R_{ci}^{-1}(\widetilde{t}_{oi} + \Delta t_{oi}) - R_{cj}^{-1}(\widetilde{t}_{oj} + \Delta t_{oj})}{||R_{ci}^{-1}(\widetilde{t}_{oi} + \Delta t_{oi}) - R_{cj}^{-1}(\widetilde{t}_{oj} + \Delta t_{oj})||_2} - \frac{t_{ij}}{||t_{ij}||_2} = \mathbf{0}, \tag{21}$$

where

$$t_{ij} = R_{ci}^{-1}t_{ci} - R_{cj}^{-1}t_{cj}. \tag{22}$$

It is approximate to solve the following linear system of equations:

$$\frac{R_{ci}^{-1}\Delta t_{oi} - R_{cj}^{-1}\Delta t_{oj}}{||t_{ij}^*||_2} + \frac{t_{ij}^*}{||t_{ij}^*||_2} - \frac{t_{ij}}{||t_{ij}||_2} = \mathbf{0}, \tag{23}$$

where

$$t^*_{ij} = R^{-1}_{ci}\widetilde{t}_{oi} - R^{-1}_{cj}\widetilde{t}_{oj}. \quad (24)$$

This is also an under-constrained equation, hence, an auxiliary condition $\Delta t_{oi} = -\Delta t_{oj}$ is introduced, considering that both the increments Δt_{oi} and Δt_{oj} at each iteration change slightly. So the final approximate solution is

$$\Delta t_{oi} = -\Delta t_{oj} = (R^{-1}_{ci} + R^{-1}_{cj})^{-1}(\frac{||t^*_{ij}||_2}{||t_{ij}||_2}t_{ij} - t^*_{ij}). \quad (25)$$

Finally, the absolute object poses are further updated by utilizing both rotation increments and translation increments. Let P^*_{oi} and P^*_{oj} represent the updated object poses. Then, we have

$$P^*_{oi} = [\widetilde{R}_{oi}\Delta R_{oi}, \widetilde{t}_{oi} + \Delta t_{oi}], \quad (26)$$

$$P^*_{oj} = [\widetilde{R}_{oj}\Delta R_{oj}, \widetilde{t}_{oj} + \Delta t_{oj}]. \quad (27)$$

3.4. Loss Function

The total loss function $\mathcal{L}_{self}$ of the proposed SMOC-Net contains three loss terms: a rotation loss $\mathcal{L}_{rot}$, a translation loss $\mathcal{L}_{trans}$ and a point matching loss $\mathcal{L}^{pair}_{pm}$.

According to the relative rotation constraint, the rotation loss is defined as

$$\mathcal{L}_{rot} = ||\omega(R_{oj}R^{-1}_{oi}R_{ij})||_2, \quad (28)$$

where $R_{ij} = R_{ci}R^{-1}_{cj}$.

According to the relative translation constraint, the translation loss is defined by measuring the angle of two relative translation vectors as

$$\mathcal{L}_{trans} = \text{acos}(t_{c,ij} \cdot t_{o,ij}), \quad (29)$$

where $\text{acos}(\cdot)$ is the arccos function and

$$t_{c,ij} = \frac{R^{-1}_{ci}t_{ci} - R^{-1}_{cj}t_{cj}}{||R^{-1}_{ci}t_{ci} - R^{-1}_{cj}t_{cj}||_2}, \quad (30)$$

$$t_{o,ij} = \frac{R^{-1}_{ci}t_{oi} - R^{-1}_{cj}t_{oj}}{||R^{-1}_{ci}t_{oi} - R^{-1}_{cj}t_{oj}||_2}. \quad (31)$$

Inspired by [15, 33, 35], the point matching loss for geometrically aligning the predicted poses and the pseudo pose labels is used:

$$\mathcal{L}_{pm} = \min_{R_o \in sym} \text{avg}_{X \in \mathcal{M}} ||(R^*_o X + t^*_o) - (R_o X + t_o)||_1, \quad (32)$$

where $||\cdot||_1$ is the L_1 norm, sym is a pool of symmetric rotations including the given rotation to deal with symmetric objects following [21], $\mathcal{M}$ represents the CAD object point cloud given in the evaluation datasets, R_o, (t_{ox}, t_{oy}) and t_{oz} are disentangled from $P_o = [R_o, t_o]$, and $P^*_o = [R^*_o, t^*_o]$ denotes the pseudo pose label. Since the proposed method deals with pairs of training patches, the sum of two point matching losses is eventually used:

$$\mathcal{L}^{pair}_{pm} = \mathcal{L}_{pm,i} + \mathcal{L}_{pm,j}. \quad (33)$$

Accordingly, the total loss function $\mathcal{L}_{self}$ is the weighted sum of the above three loss terms as

$$\mathcal{L}_{self} = \mathcal{L}_{rot} + \lambda_1\mathcal{L}_{trans} + \lambda_2\mathcal{L}^{pair}_{pm}, \quad (34)$$

where λ_1 and λ_2 are two weight parameters.

4. Experiments

4.1. Datasets

Synthetic PBR Dataset [10] As the physically-based renderer has been proven to perform better than OpenGL renderers [13, 34], we use the synthetic PBR data from [10] to train the 2D object detector Yolov4 [1] and the backbone pose estimation module GDR-Net [35].

LineMOD [8] is a standard benchmark for 6D object pose estimation, consisting of 13 sequences. And there are about 1.2k images in each sequence. As done in [13], we use 15% of images for training and the remaining for testing.

Occluded-LineMOD [2] is generated from benchvise sequence of the LineMOD dataset, and it contains 1214 images. As done in [33], we sample data for testing using the BOP split [9] and use the remaining data for training.

4.2. Metrics

We evaluate the performance using the common Average Distance of Distinguishable Model Points (ADD) metric [8], which measures the distance between 3D object model point clouds transformed by the predicted pose and the corresponding ground-truth. A pose is considered correct when the distance is less than 10% of the diameter of a 3D model. ADD-S is used [37] for symmetric objects by computing the closest point distance.

4.3. Implementation Details

Our method is implemented by Pytorch, and all experiments are conducted on an NVIDIA 3080 GPU. The camera poses are calculated using a classic SfM method COLMAP [25, 26, 36] before training. We replace ResNet-34 [7] backbone of GDR-Net [35] with ResNeSt-50 [6] backbone as done in [33]. The learning rate is set to 10^{-4}. The strategy of pre-training with synthetic data is the same as [33]. During the self-supervised training, we train the student pose estimation network for 30 epochs using un-annotated real RGB data, and the batch size is set to 8. In each epoch, all N training images are randomly divided into $N/2$ image pairs. During inference, only the student pose estimation network is used.

Training Manner	Methods	Training data	Ape	Bv.	Cam	Can	Cat	Dril	Duck	Eb.	Glue	Holep	Iron	Lamp	Phone	Mean
Fully-supervision	DPOD [41]	R	53.3	95.2	90.0	94.1	60.4	97.4	66.0	99.6	93.8	64.9	99.8	88.1	71.4	82.6
	PVNet [22]		43.6	99.9	86.9	95.5	79.3	96.4	52.6	99.2	95.7	81.9	98.9	99.3	92.4	86.3
	CDPN [17]		64.4	97.8	91.7	95.5	83.8	96.2	66.8	99.7	99.6	85.8	97.9	97.9	90.8	89.9
Self-supervision	AAE [31]	S	4.2	22.9	32.9	37.0	18.7	24.8	5.9	81.0	46.2	18.2	35.1	61.2	36.3	32.6
	MHP [20]		11.9	66.2	22.4	59.8	26.9	44.6	8.3	55.7	54.6	15.5	60.8	-	34.4	38.8
	DPOD [41]		35.1	59.4	15.5	48.8	28.1	59.3	25.6	51.2	34.6	17.7	84.7	45.0	20.9	40.5
	Self6D [34]	S+R$^-$+D	38.9	75.2	36.9	65.6	57.9	67.0	19.6	99.0	94.1	16.2	77.9	68.2	50.1	58.9
	Self6D++ [33]		75.4	94.9	97.0	99.5	86.6	98.9	68.3	99.0	96.1	41.9	99.4	98.9	94.3	88.5
	DSC-PoseNet [39]	S+R$^-$	35.9	83.1	51.5	61.0	45.0	68.0	27.6	89.2	52.5	26.4	56.3	68.7	46.4	54.7
	Self6D++ [33]		76.0	91.6	97.1	99.8	85.6	98.8	56.5	91.0	92.2	35.4	99.5	97.4	91.8	85.6
	OURS		**85.6**	**96.7**	**97.2**	**99.9**	**95.0**	**100.0**	**76.0**	**98.3**	**99.2**	**45.6**	**99.9**	**98.9**	**94.0**	**91.3**

Table 1. Comparison with state-of-the-art methods on LineMOD dataset. The table reports results for the Average Recall (%) of ADD(-S). All results except ours and DSC-PoseNet are copied from [33]. The best RGB based label-free method is marked in bold. R: annotated real RGB data. S: synthetic RGB data. R$^-$: unannotated real RGB data. D: depth data.

Methods	Training data	Ape	Can	Cat	Dril	Duck	Eb.	Glue	Holep	Mean
DPOD [41]	S	2.3	4.0	1.2	10.5	7.2	4.4	12.9	7.5	6.3
CDPN [17]		20.0	15.1	16.4	5.0	22.2	36.1	27.9	24.0	20.8
CosyPose [14]		44.0	69.9	42.1	67.5	**47.8**	24.4	60.0	17.5	46.7
Self6D [34]	S+R$^-$+D	13.7	43.2	18.7	32.5	14.4	57.8	54.3	22.0	32.1
Self6D++ [33]		59.4	96.5	60.8	92.0	30.6	51.1	88.6	38.3	64.7
DSC-PoseNet [39]	S+R$^-$	13.9	15.1	19.4	40.5	6.9	38.9	24.0	16.3	21.9
Self6D++ [33]		57.7	**95.0**	52.6	90.5	26.7	45.0	87.1	23.5	59.8
OURS		**60.0**	94.5	**59.1**	**93.0**	37.2	**48.3**	**89.3**	**25.0**	**63.3**

Table 2. Comparison with state-of-the-art methods on Occluded-LineMOD dataset.

4.4. Comparative Evaluation

Comparisons on LineMOD. Here, we evaluate the proposed SMOC-Net on the LineMOD dataset in comparison to some state-of-the-art methods, including three fully-supervised methods (DPOD [41], PVNet [22], CDPN [17]), three self-supervised methods that are trained with only synthetic data (AAE [31], MHP [20], DPOD [41]), one self-supervised method that is trained with both synthetic data and un-annotated real images (DSC-PoseNet [39]), and two self-supervised methods that are trained with synthetic data + un-annotated real images + depth images (Self6D [34], Self6D++ [33]). In addition, for a more comprehensive comparison, we also compare our method with Self6D++ [33] trained with synthetic data and un-annotated real images but without depth images. The corresponding results are reported in Table 1.

As seen from Table 1, the fully-supervised methods DPOD, PVNet, CDPN perform better than most of the existing self-supervised methods in most cases, mainly because annotated real images with object poses are used for training these fully-supervised methods. Moreover, the proposed SMOC-Net achieves a better performance than all the comparative methods (regardless of whether these methods are trained in an either fully-supervised or self-supervised manner), because of the derived relative-pose constraint and the designed camera-pose-guided refiner in SMOC-Net. These results demonstrate the effectiveness of the proposed method for 6D object pose estimation.

Comparisons on Occluded-LineMOD. We evaluate the proposed SMOC-Net on the Occluded-LineMOD dataset where there exist many occlusions, in comparison to DPOD [41], CDPN [17], CosyPose [14], Self6D [34], Self6D++ [33], and DSC-PoseNet [39]. The corresponding results are reported in Table 2. As seen from this table, the overall average recalls of most of the comparative methods are lower than 60% or even lower. Among all the comparative methods, Self6D++ trained with 3 categories of data (synthetic, unannotated real images, and depth images) performs the best, mainly because it uses additional scene depth images at the training stage, which contain 3D spatial information and are more effective in capturing the 3D spatial relationship between adjacent objects. Moreover, the proposed SMOC-Net achieves a slightly lower performance than Self6D++ that is trained with 3 categories of data (synthetic, unannotated real images, and depth images). However, the SMOC-Net significantly outperforms Self6D++ that is trained without depth images in the same manner as SMOC-Net, and it also performs significantly better than the other comparative methods. These results demonstrate that the proposed method could effectively handle the oc-

Method	Ape	Bv.	Cam	Can	Cat	Dril	Duck	Eb.	Glue	Holep	Iron	Lamp	Phone	Mean
w/o $\mathcal{L}_{rot}$	82.1	95.0	97.0	99.8	94.1	**100.0**	72.6	98.5	**99.3**	40.3	99.9	96.2	**95.2**	90.0
w/o $\mathcal{L}_{trans}$	84.3	92.9	95.9	99.8	93.9	99.7	**77.0**	**98.7**	98.7	**47.5**	**100.0**	98.1	94.3	90.8
w/o refiner*	**88.3**	96.0	96.6	99.7	92.8	**100.0**	70.4	98.0	99.0	42.2	**100.0**	97.3	94.3	90.4
OURS$^-$	50.7	**99.4**	89.5	97.2	80.0	98.7	26.3	81.1	91.2	41.9	98.8	**98.9**	64.4	77.5
OURS	85.6	96.7	**97.2**	**99.9**	**95.0**	**100.0**	76.0	98.3	99.2	45.6	99.9	**98.9**	94.0	**91.3**

Table 3. Ablation study on LineMOD dataset. Individual loss and pose refiner are evaluated on LineMOD. * : the proposed camera-pose-guided refiner is replaced with DeepIM [15]. OURS$^-$: our results trained on synthetic data. The best results are marked in bold.

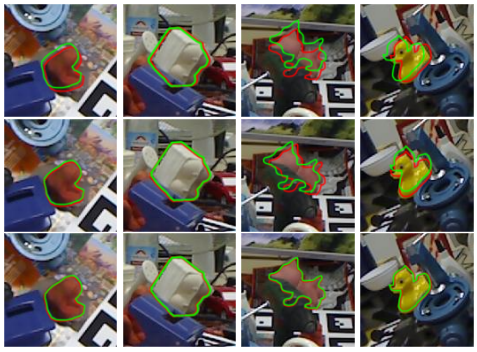

Figure 4. Visualized results on Occluded-LineMOD dataset. Top: results of DSC-PoseNet. Middle: results of Self6D++. Bottom: results of SMOC-Net. Red: the ground-truth object poses. Green: the results after self-supervised training.

clusion case. Fig. 4 illustrates the visualized results of Self6D++, DSC-PoseNet and SMOC-Net, which are all trained with synthetic data and un-annotated real images.

4.5. Ablation Study

To evaluate the effects of the losses $\mathcal{L}_{rot}$, $\mathcal{L}_{trans}$ and the camera-pose-guided refiner, we conduct the ablation study on LineMOD and the results are reported in Table 3.

Firstly, we respectively disable the rotation loss $\mathcal{L}_{rot}$ and the translation loss $\mathcal{L}_{trans}$. the corresponding models are denoted as w/o $\mathcal{L}_{rot}$ and w/o $\mathcal{L}_{trans}$ respectively. As shown in Table 3, the pose accuracy decreases when disabling each of them. These results demonstrate that both of the rotation loss and the translation loss are effective to improve the pose accuracy for our method. Then, we replace the proposed camera-pose-guided refiner with DeepIM [15], and the corresponding model is denoted as w/o refiner*. The performance of w/o refiner* is significantly lower than the complete SMOC-Net. In addition, we also train our model with only synthetic data, which is denoted as OUR$^-$. The overall average recall of OUR$^-$ is much lower than the complete SMOC-Net, demonstrating that the proposed self-supervised object pose estimation method could significantly improve the pose accuracy.

Method	Training/Rendering Time (s)	Total Time (s)
Self6D++ [33]	37/29	1110
OURS	8/-	283

Table 4. Comparison of the training times in one epoch and the total times on LineMOD.

4.6. Comparison of Training Time

Here, we compare our training time on the LineMOD dataset with Self6D++ [33], a state-of-the-art differentiable-renderer-based method.

Both the training times in one epoch and the total times by the two methods are reported in Table 4. As seen in Table 4, the training time of Self6D++ in one epoch is 37 seconds, but Self6D++ spends 29 seconds on rendering, indicating that rendering is a main time-consuming part for such a differentiable-renderer-based method. Moreover, the training time of our method in one epoch is much less than Self6D++, and the total training time (for both SfM and network training) of our method is also much less than Self6D++. These results demonstrate the priority of the proposed method in model training.

5. Conclusion

In this paper, we have proposed the SMOC-Net for self-supervised monocular object pose estimation by utilizing the predicted camera poses from un-annotated real images. Unlike most of the existing self-supervised methods that rely on differentiable renderer for providing additional constraint on object pose prediction, the SMOC-Net utilizes the derived relative-pose constraint for boosting the performance on object pose estimation. We also propose a pose refiner for further refining the initial object pose with the derived relative-pose constraint. The experimental results on two public datasets demonstrate the effectiveness of the proposed method even when dealing with strong occlusions.

Acknowledgements. This work was supported by the National Natural Science Foundation of China (Grant Nos. U1805264 and 61991423), the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDB32050100), the Beijing Municipal Science and Technology Project (Grant No. Z211100011021004).

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv preprint arXiv:2004.10934*, 2020. 3, 6

[2] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6D Object Pose Estimation Using 3D Object Coordinates. In *ECCV*, pages 536–551, 2014. 6

[3] Avishek Chatterjee and Venu Madhav Govindu. Robust Relative Rotation Averaging. *PAMI*, 40(4):958–972, 2017. 5

[4] Hansheng Chen, Pichao Wang, Fan Wang, Wei Tian, Lu Xiong, and Hao Li. EPro- PnP: Generalized end-to-end probabilistic perspective-n-points for monocular object pose estimation. In *CVPR*, pages 2781–2790, 2022. 2

[5] Qiulei Dong, Xiang Gao, Hainan Cui, and Zhanyi Hu. Robust Camera Translation Estimation via Rank Enforcement. *IEEE Tran. on Cybernetics*, 52(2):862–872, 2022. 4

[6] Zhang Hang, Wu Chongruo, Zhang Zhongyue, Zhu Yi, Lin Haibin, Zhang Zhi, Sun Yue, He Tong, Mueller Jonas, Manmatha R., Li Mu, and Smola Alexander. ResNeSt: Split-Attention Networks. In *CVPR*, pages 2735–2745, 2022. 6

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, pages 770–778, 2016. 6

[8] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes. In *ACCV*, pages 548–562, 2012. 2, 6

[9] Tomas Hodan, Frank Michel, Eric Brachmann, Wadim Kehl, Anders Glent Buch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, Caner Sahin, Fabian Manhardt, Federico Tombari, Tae-Kyun Kim, Jiri Matas, and Carsten Rother. BOP: Benchmark for 6D Object Pose Estimation. In *ECCV*, pages 19–35, 2018. 6

[10] Tomas Hodan, Martin Sundermeyer, Bertram Drost, Yann Labbe, Eric Brachmann, Frank Michel, Carsten Rother, and Jiri Matas. BOP Challenge 2020 on 6D Object Localization. In *ECCV*, 2020. 4, 6

[11] Tomáš Hodaň, Dániel Baráth, and Jiří Matas. EPOS: Estimating 6D Pose of Objects with Symmetries. In *CVPR*, pages 11703–11712, 2020. 2

[12] Shun Iwase, Xingyu Liu, Rawal Khirodkar, Rio Yokota, and Kris M. Kitani. RePOSE: Fast 6D Object Pose Refinement via Deep Texture Rendering. In *ICCV*, pages 3283–3292, 2021. 2

[13] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *ICCV*, pages 1521–1529, 2017. 2, 6

[14] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. CosyPose: Consistent multi-view multi-object 6D pose estimation. In *ECCV*, pages 574–591, 2020. 7

[15] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. DeepIM: Deep Iterative Matching for 6D Pose Estimation. *IJCV*, pages 1–22, 2019. 5, 6, 8

[16] Zhigang Li, Yinlin Hu, Mathieu Salzmann, and Xiangyang Ji. SD-Pose: Semantic Decomposition for Cross-Domain 6D Object Pose Estimation. In *AAAI*, pages 2020–2028, 2021. 1, 2

[17] Zhigang Li, Gu Wang, and Xiangyang Ji. Cdpn:Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *ICCV*, pages 7678–7687, 2019. 1, 2, 7

[18] Huang Lin, Hodan Tomas, Ma Lingni, Zhang Linguang, Tran Luan, Twigg Christopher, Wu Po-Chen, Yuan Junsong, Keskin Cem, and Wang Robert. Neural Correspondence Field for Object Pose Estimation. In *ECCV*, pages 585–603, 2022. 2

[19] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft Rasterizer: A Differentiable Renderer for Image-based 3D Reasoning . In *ICCV*, pages 7708–7717, 2019. 1

[20] Fabian Manhardt, Diego Martin Arroyo, Christian Rupprecht, Benjamin Busam, Tolga Birdal, Nassir Navaband, and Federico Tombari. Explaining the Ambiguity of Object Detection and 6D Pose From Visual Data. In *ICCV*, pages 6841–6850, 2019. 7

[21] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *ICCV*, pages 7668–7677, 2019. 2, 6

[22] Sida Peng, Xiaowei Zhou, Yuan Liu, Haotong Lin, Qixing Huang, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *CVPR*, pages 4561–4570, 2019. 1, 2, 7

[23] Jason Rambach, Chengbiao Deng, Alain Pagani, and Didier Stricker. Learning 6DoF Object Poses from Synthetic Single Channel Images. In *IEEE ISMAR*, 2018. 2

[24] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for Data: Ground Truth from Computer Games. In *ECCV*, pages 102–118, 2016. 1

[25] Johannes L. Schonberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *CVPR*, pages 4104–4113, 2016. 2, 4, 6

[26] Johannes L. Sch¨onberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *ECCV*, pages 501–518, 2016. 2, 4, 6

[27] Juil Sock, Guillermo Garcia-Hernando, Anil Armagan, and Tae-Kyun Kim. Introducing Pose Consistency and Warp-Alignment for Self-Supervised 6D Object Pose Estimation in Color Images. In *3DV*, pages 291–300, 2020. 3

[28] Chen Song, Jiaru Song, and Qixing Huang. Hybridpose: 6d object pose estimation under hybrid representations. In *CVPR*, pages 431–440, 2020. 2

[29] Hao Su, Charles R. Qi, Yangyan Li, and Leonidas J. Guibas. Render for CNN: Viewpoint Estimation in Images Using CNNs Trained with Rendered 3D Model Views. In *ICCV*, pages 2686–2694, 2015. 1

[30] Yongzhi Su, Mahdi Saleh, Torben Fetzer, Jason Rambach, Nassir Navab, Benjamin Busam, Didier Stricker, and Federico Tombari. ZebraPose: Coarse to Fine Surface Encoding for 6DoF Object Pose Estimation. In *CVPR*, pages 6738–6748, 2022. 2

[31] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *ECCV*, pages 699–715, 2018. 1, 2, 7

[32] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua. Real-Time Seamless Single Shot 6D Object Pose Prediction. In *CVPR*, pages 292–301, 2018. 2

[33] Gu Wang, Fabian Manhardt, Xingyu Liu, and Xiangyang Ji. Occlusion-Aware Self-Supervised Monocular 6D Object Pose Estimation. *PMAI*, 2021. 1, 3, 4, 6, 7, 8

[34] Gu Wang, Fabian Manhardt, Jianzhun Shao, Xiangyang Ji, Nassir Navab, and Federico Tombari. Self6D: Self-supervised Monocular 6D Object Pose Estimation. In *ICCV*, pages 108–125, 2020. 1, 3, 4, 6, 7

[35] Gu Wang, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. GDR-Net: Geometry-Guided Direct Regression Network for Monocular 6D Object Pose Estimation. In *CVPR*, pages 16611–16621, 2021. 1, 2, 3, 6

[36] Pinhe Wang, Limin Shi, Bao Chen, Zhanyi Hu, Jianzhong Qiao, and Qiulei Dong. Pursuing 3-D Scene Structures With Optical Satellite Images From Affine Reconstruction to Euclidean Reconstruction. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–14, 2022. 6

[37] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. In *Rss*, 2018. 2, 6

[38] Yan Xu, Kwan-Yee Lin, Guofeng Zhang, Xiaogang Wang, and Hongsheng Li. RNNPose: Recurrent 6-DoF Object Pose Refinement with Robust Correspondence Field Estimation and Pose Optimization. In *CVPR*, pages 14860–14870, 2022. 2

[39] Zongxin Yang, Xin Yu, and Yi Yang. DSC-PoseNet: Learning 6DoF Object Pose Estimation via Dual-Scale Consistency. In *CVPR*, pages 3907–3916, 2021. 1, 3, 4, 7

[40] Xin Yu, Zheyu Zhuang, Piotr Koniusz, and Hongdong Li. 6DoF Object Pose Estimation via Differentiable Proxy Voting Loss . In *CVPR*, 2020. 2

[41] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *ICCV*, pages 1941–1950, 2019. 2, 7