# ES6D: A Computation Efficient and Symmetry-Aware 6D Pose Regression Framework

**Ningkai Mo** [1*] **Wanshui Gan** [1,2*] **Naoto Yokoya** [2,3] **Shifeng Chen** [1†]

[1] ShenZhen Key Lab of Computer Vision and Pattern Recognition,
Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences,
[2] The University of Tokyo, [3]RIKEN
[nk.mo19941001, wanshuigan]@gmail.com, yokoya@k.u-tokyo.ac.jp, shifeng.chen@siat.ac.cn

## Abstract

*In this paper, a computation efficient regression framework is presented for estimating the 6D pose of rigid objects from a single RGB-D image, which is applicable to handling symmetric objects. This framework is designed in a simple architecture that efficiently extracts point-wise features from RGB-D data using a fully convolutional network, called XYZNet, and directly regresses the 6D pose without any post refinement. In the case of symmetric object, one object has multiple ground-truth poses, and this one-to-many relationship may lead to estimation ambiguity. In order to solve this ambiguity problem, we design a symmetry-invariant pose distance metric, called average (maximum) grouped primitives distance or A(M)GPD. The proposed A(M)GPD loss can make the regression network converge to the correct state, i.e., all minima in the A(M)GPD loss surface are mapped to the correct poses. Extensive experiments on YCB-Video and T-LESS datasets demonstrate the proposed framework's substantially superior performance in top accuracy and low computational cost. The relevant code is available in https://github.com/GANWANSHUI/ES6D.git.*

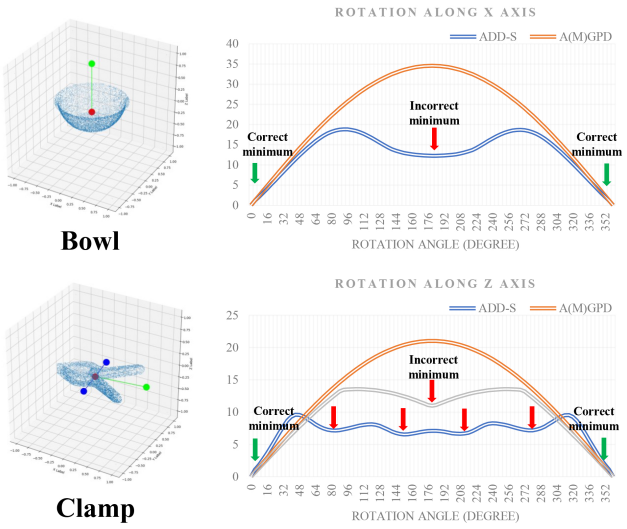


Figure 1. **Comparison of A(M)GPD and ADD-S.** Axis $X$ shows the rotation angle of the object (from 0° to 360°). Axis $Y$ shows the calculated distance. We set the initial pose as the ground truth. As we can see, all minima are mapped to correct poses in the A(M)GPD curve and several minima point to incorrect poses in the ADD-S curve.

## 1. Introduction

Estimating the 6D object pose is important for real-time applications such as augmented reality (AR) [24], autonomous driving [3, 8], and robotics [4, 34]. In recent years, methods based on the deep neural network (DNN) have gradually emerged [17, 22, 25, 26, 40]. The RGB-D-based method [35] fuses RGB features and point cloud features and shows exceptional robustness in handling heavy occlusion and textureless situations. However, as discussed below, regression methods [35, 38] will fail for some symmetric objects and its computational cost is still an obstacle for real-time application. In this paper, we propose a RGB-D-based 6D pose regression framework that is more computation efficient and applicable to symmetric objects.

Feature extraction from RGB-D data is a crucial part of our framework. The methods in [12, 19, 35] obtain robust features through a dense fusion network, which fuses RGB

*The first two authors contributed equally and should be regarded as co-first authors.
†Corresponding author.

and point cloud features with an indexing operation. However, an efficient network should avoid random memory accesses [23], which is the computational bottleneck of the dense fusion network in [12,35]. For efficiency and simplicity, a fully convolutional feature extraction network, named XYZNet, is proposed in this paper. XYZNet is much more efficient than the heterogeneous structure in [35] and [12]. The depth image is converted to the XYZ map, which is strictly aligned with the RGB image, as shown in Figure 2. Therefore, the local features from RGB and the point cloud can be simultaneously extracted with a 2D convolutional kernel. Unlike the RGB-D-based method in [21], the XYZ map is propagated to the rear layer to retain the spatial information of the local features. Then, a CNN-based PointNet [28] module is utilized to encode the point cloud with local features. Finally, the different modality features are aggregated. The experimental results reveal the superiority of the proposed XYZNet.

In addition, learning-based manners easily fail toward the symmetric object. To explain this problem, we model the network training of 6D pose estimation as minimizing the following loss:

$$l = \text{loss}(p, \hat{p}) = \text{loss}(N(I, w), \hat{p}), \tag{1}$$

where $p$ is the estimated pose from network $N(I, w)$, $\hat{p}$ is the ground truth, $I$ represents the input image, and $w$ denotes the parameters of the network. The essence of the training is constantly adjusting the parameters of the network to the direction of the gradient of $\text{loss}(p, \hat{p})$. Finally, the network will converge to the global or local minimum in the loss landscape. A symmetric object $O$ has several ground truths $S(O) = \{\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_k\}$, which are called proper symmetries of the object $O$ [1]. Typically, when using L1 loss to train the neural network of object $O$, it would converge to the state that predicts the average of $S(O)$, which is mapped to the minimum of the L1 loss surface. However, the average of $S(O)$ is meaningless.

To avoid this problem, the loss function should satisfy two requirements: (1) all minima in the loss surface are mapped to the correct poses; and (2) the loss function is continuous, as Deep Networks can only approximate the continuous functions [9, 18]. ADD-S is widely used as the loss in prior regression frameworks [33, 35, 38, 39] to handle symmetries. The ADD-S loss is always continuous but does not satisfy requirement (1) in some cases. As shown in Figure 1, several local minima in the ADD-S landscape are mapped to incorrect poses because of the particular shape of the objects. The motivation of our solution is to design a novel pose distance metric that is in the 3D metric space (meter, for instance) like ADD-S, and satisfies requirements (1) and (2). To this end, we introduce a novel shape representation for arbitrary objects named grouped primitives (GP). The GP is only associated with the proper symmetries $S(O)$ and ignores the details of the shape. Then, we divide symmetric objects into five categories and give the corresponding distance metric called the average (maximum) grouped primitives distance, or A(M)GPD. For typical symmetric objects, the validity of A(M)GPD is verified by a numerical and visualization method.

We evaluate the proposed framework on the YCB-Video [38] and T-LESS datasets [15] and demonstrate its superiority by taking into account the trade-off between speed and accuracy. In summary, the main contributions of this work are as follows.

- We propose a novel feature extraction network XYZNet for the RGB-D data, which is suitable for pose estimation with low computational cost and superior performance.
- The compact shape representation GP and the distance metric A(M)GPD are introduced to handle symmetries. The loss based on A(M)GPD can constrain the regression network to converge to the correct state.
- A numerical simulation and visualization method is carried out to analyze the validity of the A(M)GPD loss. This analytical method is applicable to other frameworks in 6D pose estimation.
- The framework ES6D is proposed by using XYZNet and the A(M)GPD loss and achieves competitive performance on the YCB-Video and T-LESS datasets.

## 2. Related Work

### 2.1. Pose estimation from RGB-D data

To make good use of the texture and geometry information of the RGB-D data, works in [11, 12, 19, 35] leverage a dense fusion network to fuse RGB and point cloud features by the indexing operation. However, the indexing operation is inefficient due to random memory access. The algorithm in [21] relates work to our network, as it also tries to extract RGB and point cloud features simultaneously with 2D convolutional kernels. However, the geometric information of the point cloud is discarded during the convolution operation, which results in lower estimation accuracy. Unlike the above methods, our framework introduces a fully convolution network, XYZNet, to obtain the point-wise features, from which poses will be regressed. Moreover, none of [11, 12, 19, 21, 35] can handle symmetries.

### 2.2. Handling symmetries in pose estimation

A symmetric object with different poses can have an identical appearance, which leads to ambiguity as described in [27]. To solve this problem, the methods in [27, 30] limit

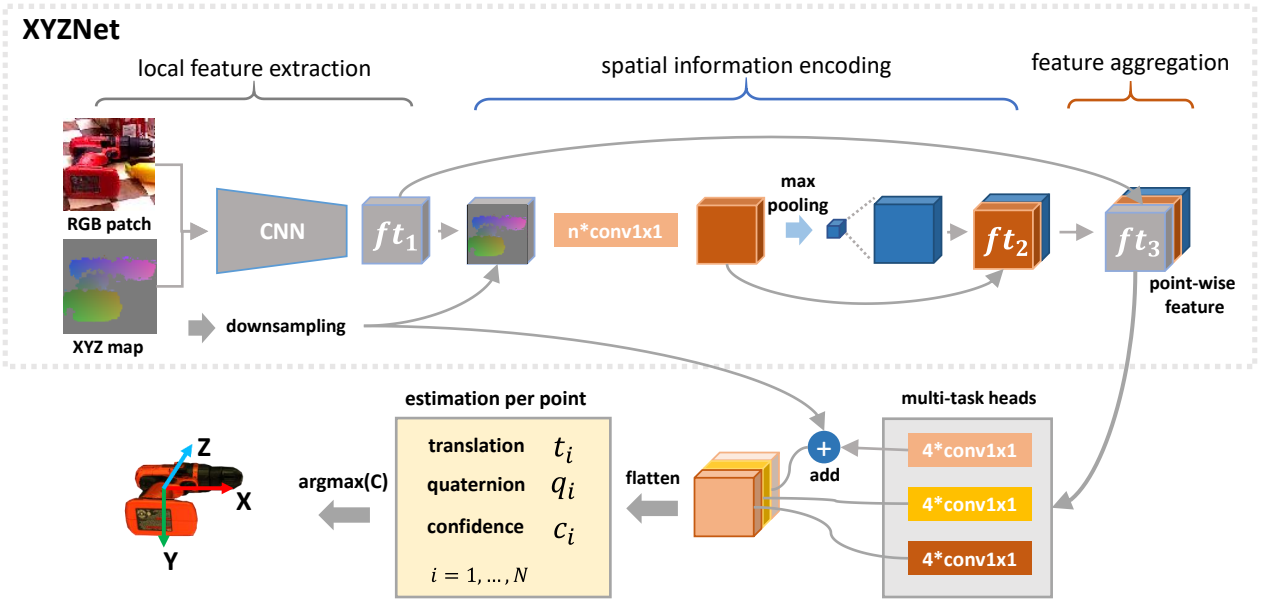


Figure 2. **Network overview.** First, the RGB-XYZ data is generated from the RGB-D image. The RGB-XYZ data is fed into a CNN module to extract local features, which encode color and geometry information. Second, the point cloud features are obtained by a PointNet-like CNN module and padded to the same size as the local features. Then, the local features and point cloud features are concatenated as the point-wise features for poses estimation. Finally, the pose with the maximum confidence is chosen as the final result.

the range of rotation in the training phase and use an additional classifier to identify the range of a rotation in the testing phase. The methods in [25,36] calculate the average distance of the corresponding pixels of all the proper symmetries $S(O)$, and choose the minimum as the final loss. The object is represented by compact surface fragments in [14], which enable the symmetries to be handled in a systematic manner. The regression methods [35, 38] avoid ambiguity by utilizing ADD-S as the loss in the training stage. The ADD-S, however, is not suitable to some symmetric objects, e.g., the bowl and large clamp in the YCB-Video dataset, as shown in Figure 1. Three ambiguity-invariant pose distance metrics ACPD, MCPD, and VSD proposed in [16] evaluate the error between the estimated pose and ground-truth pose. However, whether the surface of these metrics have incorrect minima has not been identified. Compared with the above methods, our A(M)GPD loss satisfies the following two properties at the same time: (1) all minima in the loss surface are mapped to the correct poses; and (2) the loss function is continuous.

# 3. The Proposed Method

## 3.1. Overview

The aim of this paper is to detect rigid objects and estimate the corresponding rotations $R \in SO(3)$ and translations $\boldsymbol{t} \in \mathbb{R}^3$ in the camera coordinate system from an RGB-D image. A two-stage scheme is proposed as below.

In the first stage, the segmentation network of PoseCNN [38] is utilized to obtain the mask and bounding box of the target object. Each mask and RGB-D image patch cropped by the bounding box is transmitted to the second stage.

In the second stage, a real-time framework, called ES6D, is proposed to estimate the pose. The pipeline of this framework is illustrated in Figure 2. First, the masked depth pixels are transformed into the XYZ map after normalization. Second, the XYZNet extracts the point-wise features from the concatenation of the RGB patch and XYZ map. Then, three convolution heads are utilized to predict the point-wise translation offsets, quaternions, and confidences. Finally, the pose with the maximum confidence is chosen as the final result.

## 3.2. Point-wise feature extraction

It has been verified that the point-wise feature from RGB-D data is more effective and robust than the feature from the RGB image for 6D pose estimation [12, 35]. The state-of-the-art method PVN3D [12] adopts a heterogeneous structure that obtains the point cloud features by PointNet++ [29], and then concatenates the point cloud features with the RGB features through the indexing operation. PointNet++ extracts the local features by a series of set ab-

straction layers (SAL) that groups the point cloud in a predefined search radius. However, dealing with the massive point cloud is time-consuming, and the representation ability would decrease if we cut down the set abstraction layer. One trait of the 2D convolution operation is grouping neighboring information to extract local features. Therefore, the proposed XYZNet intends to simultaneously extract the local features by doing the 2D convolution operation on the RGB-XYZ image.

First, the masked depth pixels are transformed into the point cloud $\mathcal{P} = \{(x_i, y_i, z_i)\}_{i=1}^{N}$, and then the points $P$ are translated and scaled to $[-1, 1]$ with the center of points $\boldsymbol{p}_c = \text{mean}(\mathcal{P})$ and a scale factor $\gamma$. The normalized points are denoted as $\dot{\mathcal{P}} = \{(\dot{x}_i, \dot{y}_i, \dot{z}_i)\}_{i=1}^{N}$ and formatted as an XYZ map. The strictly aligned RGB-XYZ data can be obtained by concatenating the XYZ map with the corresponding RGB patch. The method in [21] also adopts the 2D convolution network to extract point cloud features from the XYZ map, but the performance is far worse than the heterogeneous structure methods [12, 35]. The main reason for this is that the spatial information of the point cloud would be discarded when using the 2D convolution operation on the XYZ map. We design the XYZNet based on the above observations, as illustrated in Figure 2.

The XYZNet consists of three parts. **(1) Local feature extraction module**. 2D convolution layers are used to learn the local features. The different convolution kernel sizes and the downsample rates are set to enlarge the receptive field. **(2) Spatial information encoding module**. The main function of this module is to extract the point cloud features. The module concatenates the local features with the XYZ map to regain the spatial structure and utilizes the $1 \times 1$ convolution to encode the local feature and coordinate of each point. Then, the global feature is obtained by max-pooling and concatenated to each point feature to provide a global context. **(3) Feature aggregation**. The local features and point cloud features are concatenated as the point-wise features. The fusion of the two modalities makes pose estimation robust against less texture and heavy occlusion.

### 3.3. 6D pose regression

After the XYZNet is completed, the set of point-wise features $F = \{\boldsymbol{f}_i\}_{i=1}^{N}$, $\boldsymbol{f}_i \in \mathbb{R}^d$, are obtained. In this subsection, we describe how to exploit the point-wise feature $\boldsymbol{f}_i$ and the corresponding visible point $\dot{\boldsymbol{p}}_i \in \dot{\mathcal{P}}$ to estimate the rotation $R_i \in SO(3)$ and translation $\boldsymbol{t}_i \in \mathbb{R}^3$. As shown in Figure 2, three $1 \times 1$ convolution heads $(\mathcal{B}_\mathcal{T}, \mathcal{B}_\mathcal{Q}, \mathcal{B}_C)$ are adopted to regress the translation offset $\left(\Delta\dot{\boldsymbol{t}}_i \in \mathbb{R}^3\right)$, quaternion $\left(\boldsymbol{q}_i \in \mathbb{R}^4, \|\boldsymbol{q}_i\| = 1\right)$ and confidence $(c_i \in [0, 1])$.

**3D translation regression** Regarding the origin of the normalized object coordinate system as a virtual keypoint, the translation $\boldsymbol{t}_i$ can be obtained by calculating the offset $\Delta\dot{\boldsymbol{t}}_i$ between the visible point $\dot{\boldsymbol{p}}_i$ and the origin. The equation could be given as:

$$\Delta\dot{\boldsymbol{t}}_i = \mathcal{B}_\mathcal{T}\left(\boldsymbol{f}_i\right), \tag{2}$$

$$\boldsymbol{t}_i = \frac{\left(\dot{\boldsymbol{p}}_i + \Delta\dot{\boldsymbol{t}}_i\right)}{\gamma} + \boldsymbol{p}_c, \tag{3}$$

where the offset of the visible point $\dot{\boldsymbol{p}}_i$ is distributed in a specific sphere. This regression function gets a smaller output space than directly regressing the object translation [7].

**3D rotation regression** We exploit the quaternion as rotation representation following [35, 38]. We get the rotation matrix as follow:

$$R_i = Quaternion\_matrix\left(Norm\left(\mathcal{B}_Q\left(\boldsymbol{f}_i\right)\right)\right), \tag{4}$$

$$Norm(\boldsymbol{q}_i) = \frac{\boldsymbol{q}_i}{\|\boldsymbol{q}_i\|}, \tag{5}$$

where $Quaternion\_matrix(\cdot)$ denotes the function that transforms the quaternion into the rotation matrix [31].

**Confidence regression** To identify the best regression result, we set a confidence estimation head to evaluate each feature's confidence $c_i$. The equation is given as:

$$c_i = Sigmoid\left(\mathcal{B}_C\left(\boldsymbol{f}_i\right)\right). \tag{6}$$

We train the confidence branch $\mathcal{B}_C$ with the self-supervision approach that is mentioned in [35].

### 3.4. Symmetry-aware loss

The existed symmetry-invariant distance metric depends on the 3D shape of the object, such as ADD-S, ACPD, MCPD, VSD [16, 35]. However, unique shape and point-pair mismatch are the causes of incorrect minima. Besides, objects, in reality, have various shapes and we cannot guarantee these metrics are valid for every shape. Therefore, we designed grouped primitives, GP, that abstract objects of the same category into several points to avoid the uncertainty caused by the shape. Furthermore, we divide these points into groups and calculate the distance between closest points in the same group, according to Eq. 12 and 13, which avoids point-pair mismatch.

**Grouped primitives** We illustrate the pipeline of the GP construction in Figure 3. Having the 3D model of the specific object, we could calculate all symmetry axes according to Eq. 9 and 10. The primitives for grouping are composed of the endpoint of the symmetry axis and the object centroid. Specifically, the following three steps are required.

**Step 1** The basic properties of the symmetry axis-angle are defined and explained. The appearance of the object $O$ looks the same after a rotation around axis $\boldsymbol{e} = (e_x, e_y, e_z)$ by angle $\theta$. Thus, the axis $\boldsymbol{e}$ is a symmetry axis of $O$. The

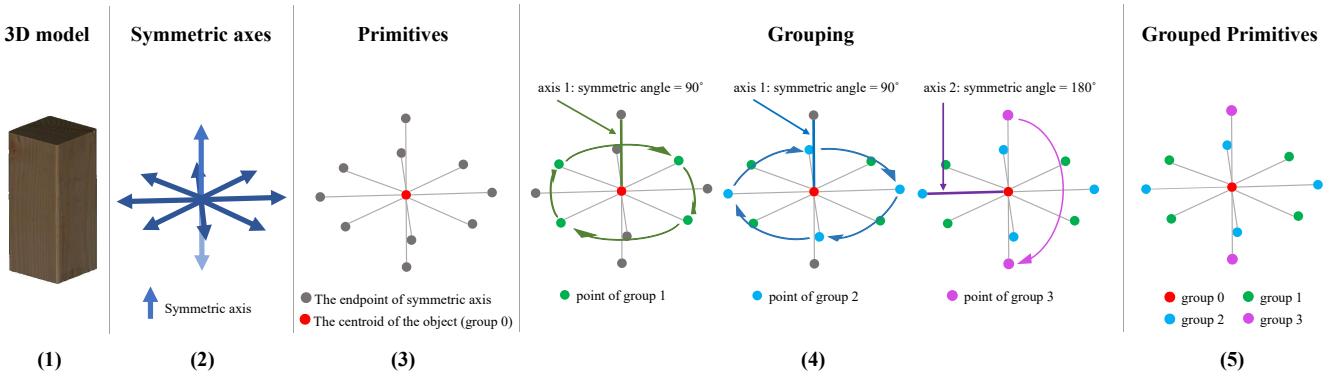


Figure 3. **The pipeline of the GP construction.**

axis $\boldsymbol{e}$ and the angle $\theta$ compose a symmetry axis-angle $\boldsymbol{a}$ , which is defined as:

$$\boldsymbol{a} = (\boldsymbol{e}, \theta), \quad \|\boldsymbol{e}\| = 1 \wedge \theta \in \{2\pi/i\}_{i=2}^{M}. \tag{7}$$

It is important to note that $2\pi$ must be an integer multiple of the angle $\theta$ of symmetry [37], and the order of $\boldsymbol{a}$ can be defined as:

$$|\boldsymbol{a}| = 2\pi/\theta(\boldsymbol{a}). \tag{8}$$

The symmetry axis-angle is a redundant form. For example, a pyramid, the object of category 2 in Figure 4, has four symmetry axis-angles: $(\boldsymbol{e}, \pi/2)$, $(\boldsymbol{e}, \pi)$, $(-\boldsymbol{e}, \pi/2)$, and $(-\boldsymbol{e}, \pi)$, where $\boldsymbol{e}$ is parallel to the green line. In this case, the four symmetry axis-angles have the same meaning for this object because of the same axis $\boldsymbol{e}$. The angles of these four symmetry axis-angles must have a greatest common divisor $\pi/2$ due to the cyclic property of rotational symmetry. **Note that, only the symmetry axis-angle, whose angle is the greatest common divisor, is used in this work**, e.g., $(\boldsymbol{e}, \pi/2)$ and $(-\boldsymbol{e}, \pi/2)$.

**Step 2** In the object coordinate system, in which the centroid of the object is used as the origin, a set of rough symmetry axis-angles of object $O$ can be obtained by using the following formula:

$$\hat{A}_O = \{\boldsymbol{a} | h\left(P_O, R(\boldsymbol{a})P_O\right) < \varepsilon\}, \tag{9}$$

where $h$ is the Hausdorff distance, $P_O$ represents the vertices of the object model, $R(\boldsymbol{a})$ is the associated rotation matrix of symmetry axis-angle $\boldsymbol{a}$, and the allowed deviation is bounded by $\varepsilon$. Then, based on symmetry axes, the Mean-Shift clustering algorithm [5] is applied to simplify $\hat{A}_O$:

$$A_O = Mean\_Shift(\hat{A}_O). \tag{10}$$

At this point, $A_O$ contains all symmetry axis-angles of the object $O$ without redundancy, where $|A_O|$ is the size of $A_O$ and is a multiple of 2 because symmetry axis-angles always come in pairs, e.g., $(\boldsymbol{e}, \pi/2)$ and $(-\boldsymbol{e}, \pi/2)$. Further, a subset $AC_O$ of $A_O$ can be obtained as:

$$AC_O = \{\boldsymbol{a} | \boldsymbol{a} \in A_O \wedge |\boldsymbol{a}| > \rho\}, \tag{11}$$

where $\rho$ is the relaxed threshold. When $|\boldsymbol{a}| > \rho$, we consider $\boldsymbol{a}$ as a continuous symmetry axis-angle, and most of the applications are covered when $\rho$ is set as 6, including all the objects to be evaluated in the experiment section. According to the size of $A_O$ and $AC_O$, symmetry objects can be divided into five categories, as shown in Figure 4.

**Step 3** As illustrated in Figure 3, if the primitive A could overlap with primitive B after a specific angle around the axis of symmetry, we regard primitive A and B lie in the same group. The grouped primitives are denoted as $G = \{g_i\}_{i=0}^{K}$, where $K$ is the size of $G$. The details of grouping principle are showed in the supplementary material.

**Pose distance metric** Based on the GP, the pose distance metric A(M)GPD is designed. The A(M)GPD contains two functions, the first of which is average grouped primitives distance (AGPD):

$$AGPD = \text{mean}_{g_i \in G} \, \text{mean}_{p_j \in g_i} \, \min_{p_k \in g_i, k \neq j} \|\hat{p}_j - \dot{p}_k\|, \tag{12}$$

where $\hat{p} = \widehat{T}p, \dot{p} = \dot{T}p, p \in g(G)$, and $\widehat{T}, \dot{T} \in SE(3)$. AGPD is used to measure the distance of two poses of object $O$, when $O$ is one of the symmetry categories $\{1, 3, 4, 5\}$ or the asymmetric object.

The category 2 is different from the others. It has only one pair of symmetry axes, which have a finite order. This property leads to an incorrect minimum in the rotation space if AGPD is used as the loss, as illustrated in the second row in Figure 1. To solve this problem, the second function maximum grouped primitives distance (MGPD) is introduced:

$$MGPD = \max_{g_i \in G} \max_{p_j \in g_i} \min_{p_k \in g_i k \neq j} \|\hat{p}_j - \dot{p}_k\|. \tag{13}$$

**Loss for regression training** The total loss of our regression framework is similar to the loss in [35], where the difference is that A(M)GPD is used to calculate the error between prediction and ground truth instead of ADD(S).

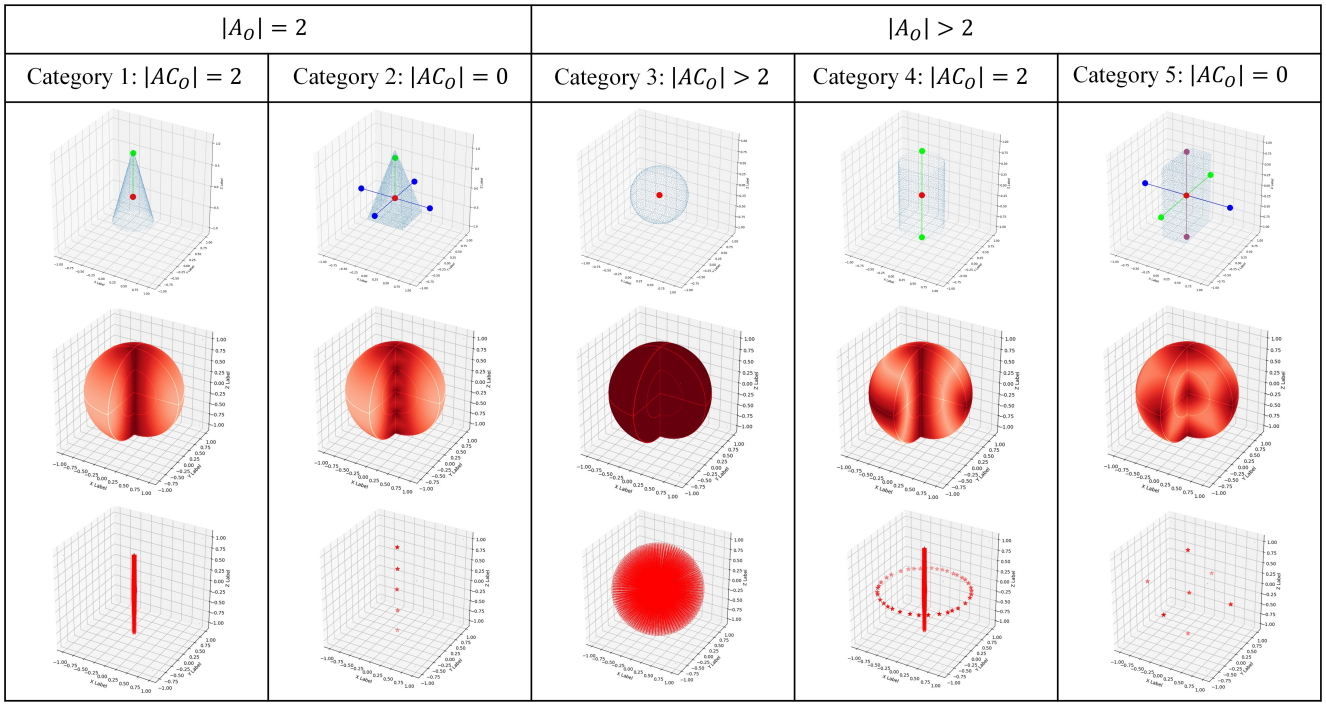


Figure 4. **Grouped primitives and the visualization of A(M)GPD landscape.** Based on the size of $A_O$ and $AC_O$, symmetric objects can be classified into five categories. For each category, a typical toy model and its grouped primitives are presented in the first row plots. The second row shows the A(M)GPD landscape of each object in the rotation space, where the darker color represents the smaller value of A(M)GPD. The third row shows the minima in each landscape. Best viewed in color.

| | | | With PoseCNN segment mask | | | | | | With GT segment mask | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FFB6D [11] | | DenseFusion (per-pixel) [35] | | DenseFusion (iterative) [35] | | ES6D | | PVN3D [12] (post process) | | ES6D | |
| | ADD-S | ADD(S) | ADD-S | ADD(S) | ADD-S | ADD(S) | ADD-S | ADD(S) | ADD-S | ADD(S) | ADD-S | ADD(S) |
| bowl | 96.3 | 96.3 | 86.0 | 86.0 | 89.5 | 89.5 | 96.4 | 96.4 | 88.7 | 88.7 | 96.8 | 96.8 |
| wood_block | 92.6 | 92.6 | 89.5 | 89.5 | 92.8 | 92.8 | 94.4 | 94.4 | 91.5 | 91.5 | 96.0 | 96.0 |
| large_clamp | 96.8 | 96.8 | 71.5 | 71.5 | 72.5 | 72.5 | 61.0 | 61.0 | 94.4 | 94.4 | 97.5 | 97.5 |
| extra_large_clamp | 96.0 | 96.0 | 70.2 | 70.2 | 69.9 | 69.9 | 59.6 | 59.6 | 91.1 | 91.1 | 96.8 | 96.8 |
| foam_brick | 97.3 | 97.3 | 92.2 | 92.2 | 92.0 | 92.0 | 96.6 | 96.6 | 96.8 | 96.8 | 96.9 | 96.9 |
| ALL | **96.6** | **92.7** | 91.2 | 82.9 | 93.2 | 86.1 | **93.6** | **89.0** | 95.7 | 91.9 | **97.1** | **93.2** |

Table 1. Comparison of 6D pose (ADD-S, ADD(S)) on the YCB-Video dataset [38]. The listed objects are symmetric. More detail could be found in the supplementary material.

### 3.5. Validation of A(M)GPD

In this subsection, a numerical and visualization method is proposed to check whether A(M)GPD meets the requirement (1) described in the introduction. In order to get a clearer view of the A(M)GPD landscape on $R \in SO(3)$, we first exploit the sampling technique to generate $N$ rotations $RC = \{R_i\}_{i=1}^{N}$ that are densely distributed on $R \in SO(3)$. Second, the identity matrix $I_{3\times3}$ is treated as the ground truth, and $\dot{R} \in RC$ is the prediction. The A(M)GPD of $I_{3\times3}$ and $\dot{R}$ can be given as $\dot{d}$,

$$\dot{d} = \text{A(M)GPD}(I_{3\times3}, \dot{R}). \tag{14}$$

Then, we visualise $\dot{d}$ with the help of the rotation vector $\boldsymbol{v} = (v_x, v_y, v_z)$, in which the direction is the rotation axis and the length is the rotation angle $\theta \in [0, \pi]$. As shown in the second row plots in Figure 4, the coordinate of $\dot{R}$ is $\boldsymbol{v}(\dot{R})$ and the color value of $\dot{R}$ is the corresponding $\dot{d}$ (the darker color represents the smaller $\dot{d}$). However, it is hard to find minima in these plots, so we further simulate the process of gradient descent by a simple algorithm. The principle of this algorithm is that $\boldsymbol{v}(\dot{R})$ constantly moves to $\boldsymbol{v}(\hat{R})$, which has the minimum $\hat{d}$ in the neighborhood of $\boldsymbol{v}(\dot{R})$ and this point will stop in a local minimum at last. We perform this principle on each $\boldsymbol{v}(\dot{R})$, and the found minima are labeled with red stars in the third row plots in Figure 4.

| | Pose Est. | $e_{ADI}$(VIVO) | $e_{VSD}$(VIVO) | $e_{ADI}$(SISO) | $e_{VSD}$(SISO) | ADD(S) | A(M)GPD | Training data | Time (s) |
|---|---|---|---|---|---|---|---|---|---|
| PointNet++ [29] | D | 0.74 | 0.50 | 0.78 | 0.54 | – | – | 37K | 0.4 |
| PPFNet [6] | D | 0.76 | 0.44 | 0.79 | 0.49 | – | – | 37K | 0.4 |
| StablePose [32] | D | **0.86** | 0.69 | **0.88** | 0.73 | – | – | 37K | 0.4 |
| Pix2Pose [25] | RGBD | – | – | – | 0.30 | – | – | 37K | 0.6 |
| CosyPose [20] | RGBD | 0.68 | 0.63 | 0.75 | 0.64 | – | – | 1M | 1.1 |
| ES6D (ADD(S)) | RGBD | 0.79 | 0.68 | 0.80 | 0.69 | 93.08 | 55.99 | 1M | **0.07** |
| ES6D (A(M)GPD) | RGBD | 0.81 | **0.75** | 0.82 | **0.76** | **93.40** | **82.70** | 1M | **0.07** |

Table 2. Comparison of 6D pose on the T-LESS dataset [15]. ES6D (ADD(S)) and ES6D (A(M)GPD) means the network is trained by the ADD(S) and A(M)GPD loss, respectively. The inference time of ES6D includes the mask segmentation cost.

As we can see, all minima are mapped to the correct poses. The other objects are presented in the supplement.

# 4. Experiments

## 4.1. Implementation detail

Our approach is implemented with Pytorch. We resize the RGB patch and XYZ map into $128 \times 128$ before putting them into the neural network. The local feature extraction module in the XYZNet is modified from ResNet18 [10]. For better performance, the grouped primitives is scaled by the object's radius. All the experiments are on an Intel (R) Xeon (R) 2.4GHz CPU with NVIDIA GTX 2080 Ti GPU.

## 4.2. Datasets

**YCB-Video** [38] is collected from 21 YCB [2] objects including 5 symmetric objects, which is a challenging task due to its various lighting conditions, significant image noise, and occlusions. The dataset contains 92 RGB-D videos, where each video shows a subset of the 21 objects in different indoor scenes. We follow prior works and split the dataset into 80 videos for training and 2,949 keyframes from the remaining 12 videos for testing. We also use the 80,000 synthetic images released by [38] in our training set.

**T-LESS** [15] is a challenging dataset with 27 symmetric objects and 3 asymmetric objects, which could effectively evaluate our proposed symmetric-aware method. Since the object is texture-less and has a similar appearance feature, it is much more challenging than the YCB-Video dataset. We use the mask result from [32] for a fair comparison.

## 4.3. Metrics

In YCB-Video dataset, following [12], the area under curve (AUC) of ADD-S and ADD(S) is treated as performance metrics for comparison of peer algorithms. In addition, the ADD(S) [13] calculates the ADD distance for non-symmetric objects and ADD-S distance for symmetric objects, which is more rigorous in evaluation than ADD-S. The AUC of ADD-S and ADD(S) for the YCB-Video dataset serve as the performance metrics.

In T-LESS dataset, we report the Average Closest Point Distance (ADI) and Visible Surface Discrepancy (VSD) following the setting in [32]. In addition, to reveal the discrepancy of ADD(S) and A(M)GPD, we compare the AUC of ADD(S) and the proposed A(M)GPD for the ablation study with the ground truth mask because the mask from [32] does not offer the index to the ground truth label.

## 4.4. Comparison with SOTA methods

**YCB-Video** To ensure a fair comparison with DenseFusion [35], we use the segmentation of PoseCNN for the testing results. Note that the large clamp and extra-large clamp in the dataset have the same appearance but with different sizes, which would cause a poor segmentation result. The failure cases in ES6D are shown in the supplemental material. From Table 1, it is observed that our method outperforms DenseFusion (iterative) by 2.9%. FFB6D [11] is better than us, which gets a better instance segmentation result by clustering after the segmentation but with an additional time cost. It is worth mentioning that no refinement and post process are used in our method, while the DenseFusion (iterative) includes the refinement and post process. In addition, we take the ground truth masks as input both in ES6D and PVN3D [12] for a comparison. In particular, our method outperforms PVN3D in symmetric objects by a large margin, *e.g.*, bowl (8.1%), wood_block (4.5%), large_clamp (3.1%), and extra_large_clamp (5.7%).

**T-LESS** Table 2 shows the comparison of 6D pose on the T-LESS dataset [15]. Pix2Pose [25] regresses pixel-wise 3D coordinates by an auto-encoder architecture. CosyPose [20] estimates the 6D pose based on the RGB image, and then does the ICP refinement with the depth image. StablePose [32] obtains 6D object pose by stable patch extraction and patch pose estimation. Compared with these methods, the proposed ES6D is a more simple and efficient framework. We achieve the best result in the VSD metric in both single instance of a single object (SISO) and varying number of instances of varying number objects in single-view RGBD images (VIVO). Besides, the inference time is much lower in comparison to these methods.

## 4.5. Ablation study

**XYZNet** We further explore the effects of the individual modules in XYZNet in Table 3. The experiments are

| Method | LEF | CXYZ | SIE | FA | A(M)GPD Loss | YCB ADD(S) | Time (ms) | FLOPs (G) | Parameters (M) |
|---|---|---|---|---|---|---|---|---|---|
| Unified_like [21] | | | | | | 91.50 | 8.4 | 7.39 | 17.85 |
| XYZNet_1 | Res18 | | | | | 91.86 | 5.1 | 7.90 | 16.29 |
| XYZNet_2 | Res18 | | √ | | | 92.04 | 5.7 | 9.16 | 17.52 |
| XYZNet_3 | Res18 | √ | √ | | | 92.42 | 5.8 | 9.16 | 17.52 |
| XYZNet_4 | Res18 | √ | √ | √ | | 93.03 | 5.9 | 10.17 | 18.51 |
| ES6D | Res18 | √ | √ | √ | √ | **93.23** | 5.9 | 10.17 | 18.51 |

Table 3. Ablation study on XYZNet. LFE: local feature extraction; CXYZ: concatenate XYZ map and local features; SIE: spatial information encoding; FA: feature aggregation. The detailed structure of each module is illustrated in Figure 2.

based on our regression framework. All methods are trained with the ADD(S) loss, except for ES6D. The experimental results demonstrate that the complete network, which comprises LEF, CXYZ, SIE, and FA, is the optimal architecture in these schemes. The Unified_like [21] structure is not satisfactory on both accuracy and inference time. Compared with XYZNet_2, XYZNet_3 obtains a large improvement by concatenating the XYZ map to local features, demonstrating the effectiveness of this explicit concatenation operation in practice. Furthermore, by adding the FA module, the XYZNet_4 yields the improvement, which illustrates the effectiveness of multimodal feature fusion (2D image and 3D point cloud).

**A(M)GPD versus ADD(S)** The motivation behind developing the A(M)GPD is the problem that the ADD(S) metric is insensitive to the rotation error of symmetry objects. For the comparison between ADD(S) loss and the proposed A(M)GPD loss, we conduct the experiment on the proposed ES6D with different loss settings. From the Table 2, it can be seen that the AUC of ADD(S) is close but there is a large gap in the A(M)GPD metric. For a more convincing result, we visualize part of the symmetric object in Figure 5. We observe that the ADD(S) loss result can have the totally reversed pose, but the ADD(S) metric can not distinguish this situation. On the other hand, we also see that the result from the A(M)GPD loss could correctly reflect this situation. By combining it with the curve illustrated in Figure 1, we can conclude that the proposed A(M)GPD loss could effectively eliminate the local minima problem in ADD-S loss during the training phase. In addition, the proposed A(M)GPD metric is much more accurate in the pose evaluation of a symmetric object.

## 5. Limitations

The performance of ES6D depends on the result of the 2D segmentation network [32, 38] and the quaternion has been proved to be discontinuous in [41]. Therefore, a unified network for instance segmentation and pose estimation, and the continuous rotation representation introduced in [41] will be investigated in future work.

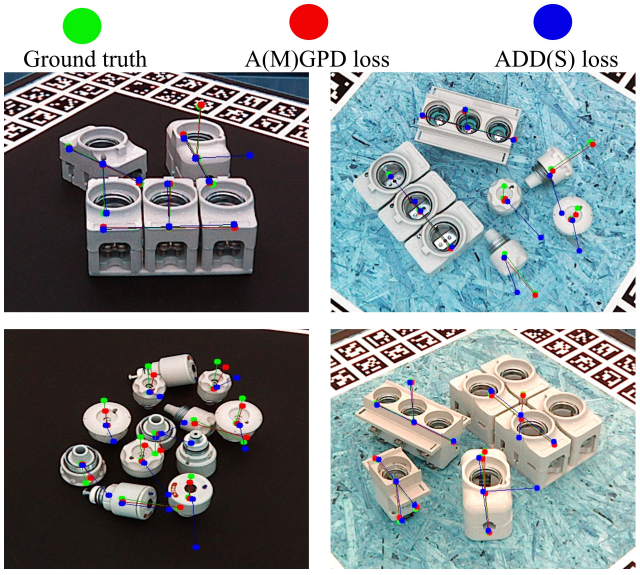


Figure 5. **Visualization on the T-LESS dataset with different training loss.** The green, red, and blue lines represent the ground truth pose, the result from A(M)GPD loss, and the result from ADD(S) loss, respectively.

## 6. Conclusion

In this paper, a novel 6D pose estimation framework, ES6D, is proposed based on the XYZNet and A(M)GPD loss. The XYZNet is designed for feature extraction from RGB-D data. It has a fully convolutional architecture and achieves an excellent trade-off between efficiency and effectiveness. Additionally, the A(M)GPD loss is proposed to handle symmetric objects, and performs better than ADD(S) loss. Moreover, a novel numerical and visualization method is introduced to check the potential incorrect suboptimal in the loss surface.

**Acknowledgments** This work is supported by Key-Area Research and Development Program of Guangdong Province (2019B010155003) and Shenzhen Science and Technology Innovation Commission (JCYJ20200109114835623).

## References


[1] Romain Brégier, Frédéric Devernay, Laetitia Leyrit, and James L. Crowley. Defining the pose of any 3d rigid object and an associated distance. *International Journal of Computer Vision*, 2018. 2

[2] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015. 7

[3] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017. 1

[4] Alvaro Collet, Manuel Martinez, and Siddhartha S Srinivasa. The moped framework: Object recognition and pose estimation for manipulation. *The international journal of robotics research*, 30(10):1284–1306, 2011. 1

[5] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002. 5

[6] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppfnet: Global context aware local features for robust 3d point matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 195–205, 2018. 7

[7] Ge Gao, Mikko Lauri, Yulong Wang, Xiaolin Hu, Jianwei Zhang, and Simone Frintrop. 6d object pose regression via supervised learning on point clouds. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3643–3649. IEEE, 2020. 4

[8] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 1

[9] Boris Hanin. Universal function approximation by deep neural nets with bounded width and relu activations. 2017. 2

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 7

[11] Yisheng He, Haibin Huang, Haoqiang Fan, Qifeng Chen, and Jian Sun. Ffb6d: A full flow bidirectional fusion network for 6d pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 2, 6, 7

[12] Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun. Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11632–11641, 2020. 1, 2, 3, 4, 6, 7

[13] 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 *Asian conference on computer vision*, pages 548–562. Springer, 2012. 7

[14] Tomas Hodan, Daniel Barath, and Jiri Matas. Epos: estimating 6d pose of objects with symmetries. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11703–11712, 2020. 3

[15] Tomáš Hodan, Pavel Haluza, Štepán Obdržálek, Jiri Matas, Manolis Lourakis, and Xenophon Zabulis. T-less: An rgb-d dataset for 6d pose estimation of texture-less objects. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 880–888. IEEE, 2017. 2, 7

[16] Tomáš Hodaň, Jiří Matas, and Štěpán Obdržálek. On evaluation of 6d object pose estimation. In *European Conference on Computer Vision*, pages 606–619. Springer, 2016. 3, 4

[17] Tomáš Hodaň, Martin Sundermeyer, Bertram Drost, Yann Labbé, Eric Brachmann, Frank Michel, Carsten Rother, and Jiří Matas. Bop challenge 2020 on 6d object localization. In *European Conference on Computer Vision*, pages 577–594. Springer, 2020. 1

[18] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991. 2

[19] Weitong Hua, Zhongxiang Zhou, Jun Wu, Huang Huang, Yue Wang, and Rong Xiong. Rede: End-to-end object 6d pose robust estimation using differentiable outliers elimination. *IEEE Robotics and Automation Letters*, 6(2):2886–2893, 2021. 1, 2

[20] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *European Conference on Computer Vision*, pages 574–591. Springer, 2020. 7

[21] Chi Li, Jin Bai, and Gregory D Hager. A unified framework for multi-view multi-class object pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 254–269, 2018. 2, 4, 8

[22] Zhigang Li, Gu Wang, and Xiangyang Ji. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7678–7687, 2019. 1

[23] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[24] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. Pose estimation for augmented reality: a hands-on survey. *IEEE transactions on visualization and computer graphics*, 22(12):2633–2651, 2015. 1

[25] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7668–7677, 2019. 1, 3, 7

[26] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4561–4570, 2019. 1

[27] Giorgia Pitteri, Michaël Ramamonjisoa, Slobodan Ilic, and Vincent Lepetit. On object symmetries and 6d pose estimation from images. In *2019 International Conference on 3D Vision (3DV)*, pages 614–622. IEEE, 2019. 2

[28] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2

[29] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017. 3, 7

[30] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3828–3836, 2017. 2

[31] Soheil Sarabandi and Federico Thomas. A survey on the computation of quaternions from rotation matrices. *Journal of Mechanisms and Robotics*, 11(2), 2019. 4

[32] Yifei Shi, Junwen Huang, Xin Xu, Yifan Zhang, and Kai Xu. Stablepose: Learning 6d object poses from geometrically stable patches. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15222–15231, 2021. 7, 8

[33] Myoungha Song, Jeongho Lee, and Donghwan Kim. Pam: Point-wise attention module for 6d object pose estimation. *arXiv preprint arXiv:2008.05242*, 2020. 2

[34] Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stan Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. *arXiv preprint arXiv:1809.10790*, 2018. 1

[35] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3343–3352, 2019. 1, 2, 3, 4, 5, 6, 7

[36] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2642–2651, 2019. 3

[37] Hermann Weyl. *Symmetry*, volume 104. Princeton University Press, 2015. 5

[38] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *Robotics: Science and Systems (RSS)*, 2018. 1, 2, 3, 4, 6, 7, 8

[39] Zelin Xu, Ke Chen, and Kui Jia. W-posenet: Dense correspondence regularized pixel pair pose regression. *arXiv preprint arXiv:1912.11888*, 2019. 2

[40] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1941–1950, 2019. 1

[41] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. On the continuity of rotation representations in neural networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 8