This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

# OVE6D: Object Viewpoint Encoding for Depth-based 6D Object Pose Estimation

Dingding Cai[1], Janne Heikkilä[2], Esa Rahtu[1]
[1]Tampere University, [2]University of Oulu
{dingding.cai, esa.rahtu}@tuni.fi, janne.heikkila@oulu.fi

## Abstract

*This paper proposes a universal framework, called OVE6D, for model-based 6D object pose estimation from a single depth image and a target object mask. Our model is trained using purely synthetic data rendered from ShapeNet, and, unlike most of the existing methods, it generalizes well on new real-world objects without any fine-tuning. We achieve this by decomposing the 6D pose into viewpoint, in-plane rotation around the camera optical axis and translation, and introducing novel lightweight modules for estimating each component in a cascaded manner. The resulting network contains less than 4M parameters while demonstrating excellent performance on the challenging T-LESS and Occluded LINEMOD datasets without any dataset-specific training. We show that OVE6D outperforms some contemporary deep learning-based pose estimation methods specifically trained for individual objects or datasets with real-world training data. The implementation is available at https://github.com/dingdingcai/OVE6D-pose.*

## 1. Introduction

The 6D pose of an object refers to a geometric mapping from the object coordinate system to the camera reference frame [18, 22]. Most commonly, this transformation is defined in terms of 3D rotation (object orientation) and 3D translation (object location). The ability to infer the object pose is an essential feature for many applications interacting with the environment. For instance, in robotic manipulation [10] and augmented reality [30], the pose is needed for grasping or realistically rendering artificial objects.

In recent works, the object pose estimation problem is commonly approached by either establishing local correspondences between the object 3D model and the observed data [16, 17, 36], or via direct regression [6, 39]. In both cases, the inference models are often optimized and stored separately for each object instance. Such approach quickly turns intractable as the number of object instances grows. Meanwhile, some existing works [49, 59] consider building a single model for multiple objects. However, to retain the performance, the model requires expensive re-training every time a new object instance is added to the database. In addition, most of the best-performing methods need annotated real-world training data, which is laborious to obtain. Although some works [24, 41, 43] consider using synthetic examples in training, they suffer from noticeable performance degradation due to the domain gap.

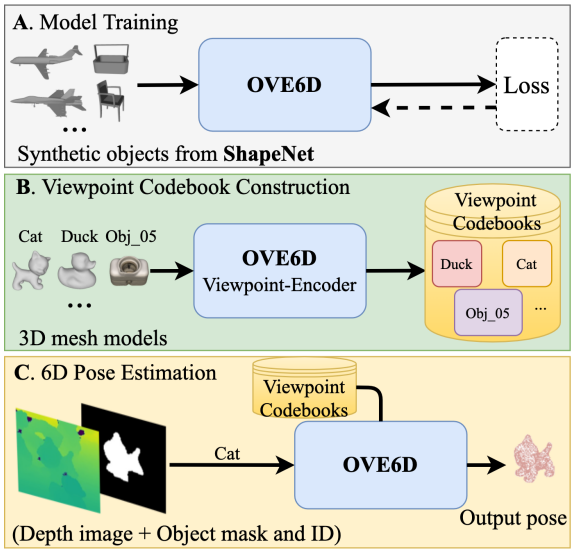
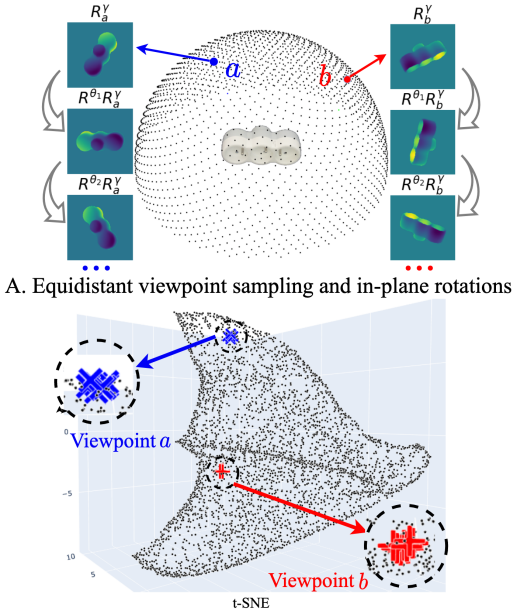



Figure 1. **A)** We propose a single universal pose estimation model (called OVE6D) that is trained using more than 19,000 synthetic objects from ShapeNet. **B)** The pre-trained model is applied to encode the 3D mesh model of the target object (unseen during the training phase) into a viewpoint codebook. **C)** At the inference time, OVE6D takes a depth image, an object segmentation mask, and an object ID as an input, and estimates the 6D pose of the target object using the corresponding viewpoint codebook. New object can be added by simply encoding the corresponding 3D mesh model and including it into the codebook database (**B**).

An alternative approach, called LatentFusion, was proposed in [32]. In this work, they first reconstruct a latent 3D object model from a small set of reference views, and later use the model to infer the 6D pose of the corresponding object from the input image. The main advantage is the ability




A. Equidistant viewpoint sampling and in-plane rotations

B. Visualisation of viewpoint embeddings

Figure 2. **A)** 4,000 viewpoints uniformly sampled from a sphere centered on the object (only the upper hemisphere is shown). The in-plane rotations $R^{\theta_i}$ around the camera optical axis are illustrated by synthesizing three examples at viewpoints $a$ ($R_a^\gamma$) and $b$ ($R_b^\gamma$). **B)** The illustration of the proposed viewpoint embeddings using t-SNE [47], where the blue ('x') and red ('+') points correspond to the embeddings from 10 in-plane rotated views at the viewpoints $a$ and $b$, respectively, and the black points represent the remaining viewpoints. It can be observed that the embeddings are relatively invariant to the in-plane rotations while varying with respect to the camera viewpoint.

to add new objects by simply generating new latent models while keeping all network parameters fixed. However, the method is computationally expensive as it is based on iterative optimization at the inference time. Furthermore, LatentFusion is very sensitive to occlusions in the input data, resulting in a significant drop in performance.

In this paper, we present a new approach, called OVE6D, for estimating the 6D object pose from a single depth image and the object segmentation mask. We further assume to have access to the 3D mesh model of the target object. Similar to LatentFusion, our approach generalizes to new objects without any re-training of model parameters. Moreover, unlike LatentFusion, the proposed method is computationally efficient and robust to occlusions in the input data. In fact, OVE6D obtains the new state-of-the-art results on the challenging T-LESS dataset [21], surpassing even approaches optimized particularly for this dataset.

The proposed approach consists of three stages as illustrated in Figure 1. First (Fig. 1 A), we train the model parameters using a large number of synthetic 3D object models from the ShapeNet [4] dataset. This stage is performed only *once* and the resulting parameters remain fixed in later stages. Second (Fig. 1 B), we convert the 3D mesh models of the target objects into viewpoint codebooks. The conversion is performed *once for each object* and it takes roughly 30 seconds per instance. Finally (Fig. 1 C), the 6D pose is inferred from the input depth image and object segmentation mask. The complete OVE6D model contains less than 4M parameters and requires roughly 50ms to infer the pose for a single object. New, previously unseen, object can be added by simply encoding the corresponding 3D mesh model as in the second stage.

The core of OVE6D is a depth-based object viewpoint encoder that captures the object viewpoint into a feature vector. The encoded representations are trained to be invariant to the in-plane rotation around the camera optical axis, but to be sensitive to the camera viewpoint, as illustrated in Figure 2. At the inference time, we first utilize the viewpoint encodings to determine the camera viewpoint, and subsequently estimate the remaining pose components (camera in-plane rotation and object 3D position) conditioned on the obtained viewpoint. The cascaded pipeline allows compact architectures for each sub-task and enables efficient training using thousands of synthetic objects.

To summarize, our key contributions are: 1) We propose a cascaded object pose estimation framework, which generalizes to previously unseen objects without additional parameter optimization. 2) We propose a viewpoint encoder that robustly captures object viewpoint while being insensitive to the in-plane rotations around the camera optical axis. 3) We demonstrate the new state-of-the-art results on T-LESS [21], without using any images from the dataset to train our model.

## 2. Related Work

**Pose estimation from RGB data** Most RGB-based object 6D pose estimation methods [1,20,33,35,36,38,44,57] attempt to establish sparse or dense 2D-3D correspondences between the 2D coordinates in the RGB image and the 3D coordinates on the object 3D model surface. The 6D pose is computed by solving the Perspective-n-Point (PnP) problem [28]. These methods achieve impressive performance for objects with rich textures providing sufficient local features to determine reliable 2D-3D correspondences. Another intuitive way to estimate the 6D pose is to directly predict the pose parameters using regression or classification, such as [3,24,46,54]. Most of these methods are based on supervised learning and rely on real-world training data with pose annotations. However, recent self-supervised approaches [40–42,50] take full advantage of the costless synthetic data for training and perform 6D object pose estimation in real-world images at testing time. Similarly, we also adopt self-supervised learning in our work and purely train our network on synthetic data.

**Pose estimation from depth data** Some deep learning-based 6D object pose estimation methods use depth-only data. Gao *et al*. [14] proposed CloudPose, which is known as the first deep learning system that performs 6D pose regression from the point cloud segments created from the object depth image. Later, Gao *et al*. proposed CloudAAE [13] to improve the generalization of the network trained on synthetic depth data by adopting an augmented autoencoder (AAE) [43] point cloud based architecture. They argue that the domain gap between the synthetic and the real images is considerably smaller and easier to fill for depth information. Bui *et al*. [2] proposed a multi-task framework combining manifold learning and 3D orientation regression directly from depth images to learn view descriptors. It was further leveraged to either retrieve or regress the 3D pose. Bui *et al*. [2] is most similar to our work, but in this work, we decouple the complete 3D orientation into the viewpoint (out-of-plane rotation) for retrieval and the 2D in-plane rotation for regression. The recent method StablePose [39] adopts the geometric stability analysis of object patches and directly predicts the patch poses in a stable group to further determine the pose of the object, which achieves state-of-the-art performance on the T-LESS dataset.

**Pose estimation from RGB-D data** When both RGB images and depth images are available, the most straightforward utilization of the two modalities is to first perform the initial pose estimation based on RGB images and then further refine with depth images, such as via ICP refinement [19, 41, 42, 54]. Alternatively, the 2D-3D feature fusion-based approaches [6,16,17,49,55] directly fuse the deep appearance features and the deep geometry features extracted from RGB and depth data by deep neural networks. These methods take full advantage of multi-modal inputs and have achieved high performance on benchmark datasets. Kehl *et al*. [25] adopt RGB-D patch descriptors extracted by CNN for 6D pose vote casting, which ignores the holistic structures of objects and easily suffers from the poor local textures. In contrast, template-based methods [52, 56] employ triplet loss to learn view descriptors from entire RGB-D images for object recognition and 3D pose estimation via nearest neighbor search.

**Pose estimation for untrained objects** Many category-level 6D object pose estimation methods [5, 7, 8, 45, 51] have appeared recently and shown good generalization to untrained objects within the same category by assuming that the same canonical pose and similar shape are shared for all instances within a category. In contrast, this assumption is not required by the instance-level object-agnostic methods LatentFusion [32] and MP-Encoder [41]. The network in LatentFusion [32] is completely trained on the synthetic data rendered from ShapeNet [4], and it is capable of generalizing to different real-world datasets when a few reference RGB images with pose annotations are provided at testing time. We follow LatentFusion [32] to train our network on the ShapeNet. However, OVE6D does not require any pose annotation at the training or testing time. In addition, DeepIM [29] can perform pose refinement for the untrained objects when the initial pose is given.

## 3. Method

In this section, we present our framework called, OVE6D, for 6D object pose estimation. Here, we assume that object IDs are known, 3D mesh models are available and object segmentation masks are provided. The task is to predict a rigid transformation from the object coordinate system to the camera coordinate system. Such transformation can be represented with a rotation $R \in SO(3)$ and a translation $t \in R^3$. The rotation $R$ can be further factorized into the out-of-plane rotation (viewpoint) $R_\gamma$ and the in-plane orientation (rotation around the camera optical axis) $R_\theta$, *i.e*., $R = R_\theta R_\gamma$, (see Fig. 2A). More details are provided in the supplementary material.

### 3.1. Overview

The OVE6D framework is illustrated in figures 1, 3 and 4. In the training phase, the model parameters are optimized using the synthetic 3D objects from ShapeNet [4]. Next, the object viewpoint codebooks are constructed with the viewpoint encoder module (see Fig. 5). At the inference time, we perform the following subtasks in a cascaded fashion. First, an initial location estimate is computed using the input depth image and the object segmentation mask, and applied to preprocess the depth image (see Fig. 3A). Second, we retrieve multiple viewpoint candidates from the object viewpoint codebook (see Fig. 3B). Third, we regress the in-plane 2D rotation with respect to each retrieved viewpoint candidate and obtain a set of complete 3D orientation estimates (see Fig. 3C). Next, we calculate a consistency score for each orientation hypothesis and output one (or more) estimate according to the score values (see Fig. 3D). Finally, the initial location estimate is refined based on the obtained 3D orientation (see Fig. 3E). The following subsections outline further details of the model components and the training procedures.

### 3.2. Preprocessing

First, we calculate and subtract the median distance $d_c$ from the segmented input depth image $D_M$ (obtained by element-wise multiplication of the depth image and the segmentation mask $M$). Next, we calculate the center coordinate $(c_x, c_y)$ of the bounding box enclosing the input segmentation mask, and form an initial estimate of the object 3D location as $t^{init} = K^{-1}[c_x, c_y, d_c]^T$, where $K$ is

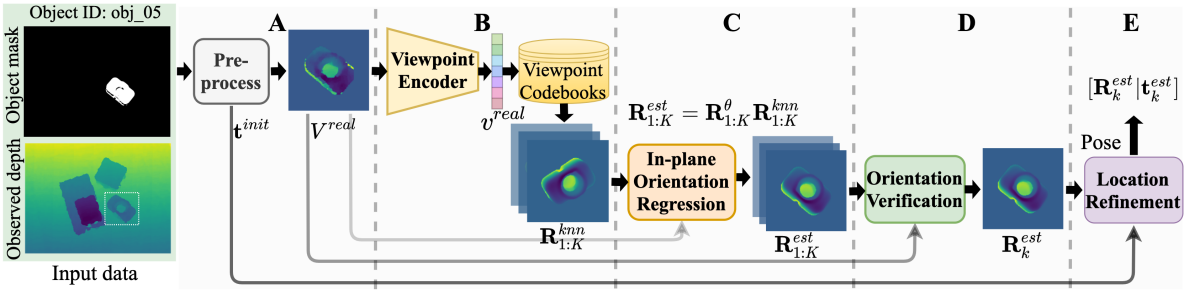



Figure 3. **The inference pipeline of OVE6D**. The entire system operates in a cascaded manner. First, the raw depth image is pre-processed to $128 \times 128$ input (**A**). Second, the object orientation is obtained by performing the viewpoint retrieval (**B**), in-plane orientation regression (**C**), and orientation verification (**D**). Finally, the object location is refined (**E**) using the obtained orientation and the initial location (**A**).

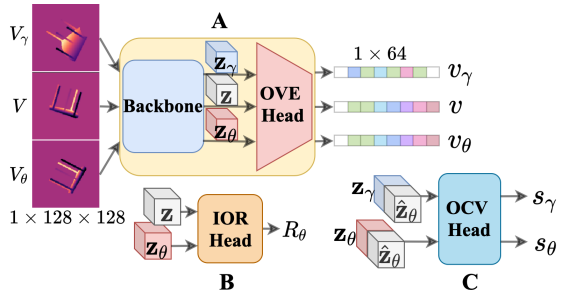



Figure 4. **Training the networks**. The proposed model contains three sub-networks to be trained. The feature maps $(\mathbf{z}, \mathbf{z}_\theta, \mathbf{z}_\gamma)$ are first extracted from the rendered depth images by the shared backbone network, and then consumed by the Object Viewpoint Encoder (**OVE**) head (**A**), the In-plane Orientation Regression (**IOR**) head (**B**), and the Orientation Consistency Verification (**OCV**) head (**C**). $\hat{\mathbf{z}}_\theta$ is transformed from $\mathbf{z}$ with the rotation $R_\theta$.

the camera intrinsic matrix. Finally, we follow LatentFusion [32] to re-scale and crop $D_M$, according to the estimated location $t^{init}$ to produce $128 \times 128$ pre-processed input depth image for the later stages.

### 3.3. Object Viewpoint Encoder

The viewpoint encoder is a lightweight neural network comprising of a CNN-based backbone (eight Conv2D + BN layers) and an encoder head $F_{OVE}$ (a single Conv2D, Pooling, and FC layer). The encoder takes the preprocessed $128 \times 128$ depth image as input and outputs a feature vector with 64 elements. The feature vector is intended to encode the camera viewpoint, but to be invariant to the in-plane rotation around the camera optical axis.

We train the viewpoint encoder using depth images rendered from ShapeNet [4]. The generated samples are organized into triplets $\{V, V_\theta, V_\gamma\}$, where $V$ and $V_\theta$ differ only in terms of in-plane rotation (by angle $\theta$), and $V_\gamma$ is rendered from a different camera viewpoint (by angle $\gamma$). The depth images are further embedded into feature representations $\{v, v_\theta, v_\gamma\}$ using the viewpoint encoder network (see Fig. 4A). The encoder parameters are optimized to rank the representation pairs according to the cosine similarity, *i.e.*, $S(v, v_\theta) > S(v, v_\gamma)$, where $S$ is the cosine similarity function. Thus, the equivalent loss function can be written as,

$$\ell^{vp} = \max(S(v, v_\gamma) - S(v, v_\theta) + m^{vp}_\lambda, 0), \qquad (1)$$

where $m^{vp}_\lambda \in (0, 1)$ is the ranking margin.

The trained viewpoint encoder is later used to construct viewpoint codebooks for novel real-world objects. To do this, we first uniformly sample $N$ viewpoints $\{R_i\}^N_{i=1}$ from a sphere centered on the object with the radius of $d_{radius} = f_{base} * d_{diameter}$, where $d_{diameter}$ is the object diameter (obtained from the 3D mesh model) and $f_{base}$ is a distance factor (5 in this paper). Then the synthetic noise-free depth images $\{V^{syn}_i\}^N_{i=1}$ are rendered using the sampled viewpoints and the object 3D mesh model. Last, viewpoint representations $\{v_i\}^N_{i=1}$ are extracted from these images (preprocessed as described in Sec. 3.2) using the viewpoint encoder and stored into the codebook database along with the object mesh model, as illustrated in Figure 5. The generated viewpoint codebook is a set $\{\{v_i, R_i\}^N_{i=1}, O_{mesh}, O_{id}\}$ that contains the corresponding viewpoint embeddings, rotation matrices, mesh model, and object ID. The entire construction requires approximately 30 seconds per object with $N = 4000$ viewpoint samples.

At the inference time, the object viewpoint representation $v^{real}$ is first extracted from the preprocessed depth image $V^{real}$ using the viewpoint encoder. Then, we utilize $v^{real}$ to compute the cosine similarity scores with all entries in the corresponding viewpoint codebook (indexed with the known object ID). The entry $\{v^{knn}, R^{knn}\}$ with the highest similarity between $v^{real}$ and $v^{knn}$ is selected as the nearest viewpoint for $V^{real}$. Optionally, we can select multiple candidate entries $\{v^{knn}_k, R^{knn}_k\}^K_{k=1}$ from the codebook to obtain a pool of $K$ viewpoint hypotheses according to the descending cosine similarity scores, as shown in Figure 3B.

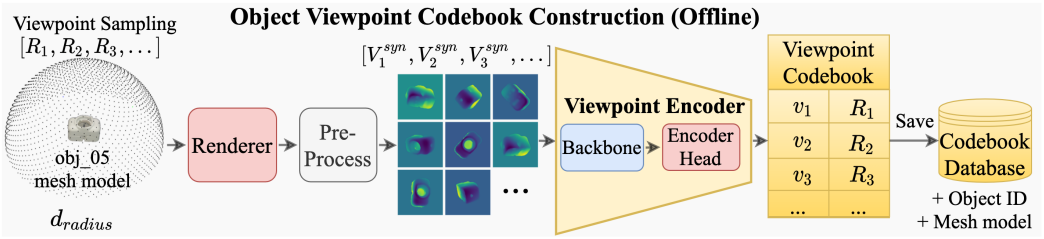



Figure 5. **Viewpoint codebook construction.** The viewpoints are sampled from a sphere centered on the object mesh model with a radius proportional to the object diameter. The viewpoint representations are extracted from the rendered depth images by the viewpoint encoder.

### 3.4. In-plane Orientation Regression

Once the viewpoint is known, the in-plane rotation around the camera optical axis can be approximated using a 2D rotation of the depth image (exact for orthographic camera). To this end, we build a regression network by appending a regression head $F_{IOR}$ (one Conv2D and two consecutive FC layers) to the backbone shared with the viewpoint encoder. This module takes a pair of feature maps $\{\mathbf{z}, \mathbf{z}_\theta\} \in R^{c\times h\times w}$ of the same viewpoint with varying in-plane orientations (intra-viewpoint) as input and regresses the relative in-plane rotation angle $\theta$ (represented as a matrix $R_\theta$), as shown in Figure 4B.

We train this module to minimize the discrepancy between the depth images transformed by the ground-truth rotation matrix $\hat{R}_\theta$ and the predicted $R_\theta$. Here, we employ a negative logarithmic cosine similarity to measure the discrepancy, written as

$$\begin{aligned} S_{cos} &= S(\mathbb{F}(T_{R_\theta}(V)),\ \mathbb{F}(T_{\hat{R}_\theta}(V))), \\ \ell^\theta &= -\log((1.0 + S_{cos}) \ / \ 2.0), \end{aligned} \tag{2}$$

where $\mathbb{F}$ refers to the flattening operation, $T_{R_\theta}$ represents the 2D spatial transformation [23] with $R_\theta$, and $V$ is the viewpoint depth image.

At the inference time, we first use the shared backbone network to extract a feature map pair $\{\mathbf{z}^{real}, \mathbf{z}_k^{knn}\}$ from the preprocessed depth image pair $\{V^{real}, V_k^{knn}\}$, where $V_k^{knn}$ is the synthesized depth image using the retrieved viewpoint $R_k^{knn}$. Next, the regression module takes the feature map pair to estimate the relative 2D rotation matrix $R_k^\theta = F_{rot}(\mathbf{z}^{real}, \mathbf{z}_k^{knn})$ to produce the complete 3D orientation estimate via $R_k^{est} = R_k^\theta R_k^{knn}$. In addition, the in-plane orientation regression can be concurrently performed for several retrieved viewpoints to obtain multiple 3D orientation hypotheses $\{R_k^{est}\}_{k=1}^K$, as shown in Figure 3C.

### 3.5. Orientation Consistency Verification

Multiple complete 3D orientation hypotheses $\{R_k^{est}\}_{k=1}^K$ can be derived from the previous modules, as described in Section 3.4. To rank the candidates, we adopt an orientation verification module that estimates the consistency between the candidates and the actual object orientation depicted in $V^{real}$. Similar to the regression module, the verification module is built by appending a verification head $F_{OCV}$ (two Conv2D layers, a Pooling and FC layer) to the shared backbone.

At the training time, we adopt a ranking-based loss to optimize this module. As shown in Figure 4C, the feature map $\mathbf{z}$ is first spatially transformed using the in-plane rotation, *i.e.* $\hat{\mathbf{z}}_\theta = T_{R_\theta}(\mathbf{z})$ where $T_{R_\theta}$ is the spatial transformation [23] with $R_\theta$. Then, we separately concatenate $\hat{\mathbf{z}}_\theta$ with $\mathbf{z}_\gamma$ and $\mathbf{z}_\theta$ along the feature channel dimension, *i.e.*, $[\hat{\mathbf{z}}_\theta; \mathbf{z}_\gamma]$ and $[\hat{\mathbf{z}}_\theta; \mathbf{z}_\theta]$, where $[;]$ denotes the concatenation, and feed them into $F_{OCV}$ to produce the consistency scores $s_\gamma$ and $s_\theta$. The equivalent loss function can be written as,

$$\ell^{css} = \max(s_\gamma - s_\theta + m_\lambda^{css}, 0), \tag{3}$$

where $m_\lambda^{css} \in (0, 1)$ is the ranking margin.

During the inference, we transform the feature map $\mathbf{z}_k^{knn}$, from the retrieved viewpoint, using the estimated in-plane rotation $R_k^\theta$ and feed it to the verification head $F_{css}$ along with the feature map $\mathbf{z}^{real}$ from the observed depth image, as shown in Figure 3D. In this way, we obtain a consistency score for each 3D orientation hypothesis. According to the estimated scores, we rank all hypotheses $\{R_k^{est}\}_{k=1}^K$ in descending order and select the top $P \in [1, K]$ orientation proposals $\{R_p^{est}\}_{p=1}^P$ as the output.

### 3.6. Location Refinement

We further refine the initial location estimate $t^{init}$ based on the obtained 3D orientation. Specifically, we first synthesize a depth image $D_p^{est}$ using the object mesh model and the pose $[R_p^{est}|t^{init}]$, where $R_p^{est}$ is the 3D orientation obtained in Section 3.5. Next, we estimate the 3D centroid $t_p^{syn}$ of the object in the depth image $D_p^{est}$, as described in Section 3.2. Furthermore, we calculate an offset $t'_\Delta = t^{init} - t_p^{syn}$ which can be regarded as the position offset caused by the self-occlusion of the object in the current orientation $R_p^{est}$. We assume that $t'_\Delta$ is approximately equal to $t_\Delta = t^{est} - t^{init}$, which allows us to obtain the final 3D location estimate $t_p^{est} = 2t^{init} - t_p^{syn}$ for the object with the

$p^{th}$ orientation proposal, as $t_p^{est} - t^{init} = t^{init} - t_p^{syn}$.

### 3.7. Pose Hypothesis Selection and Refinement

As presented in previous sections, we may obtain multiple orientation proposals, each of which results in one pose hypothesis. We calculate the following quality measure for each pose hypothesis,

$$q_p = \frac{1}{m_p} \sum \mathbb{I}(|D_p^{syn} - D_M| > 0.1d) \tag{4}$$

where $\mathbb{I}$ represents the indicator function, $D_M$ is the segmented object depth image (obtained in Sec. 3.2), $D_p^{syn}$ is the rendered depth image with the pose hypothesis $\{R_p^{est}|t_p^{est}\}$, $m_p$ is the total number of pixels belonging to the object in $D_p^{syn}$, and $d$ is the diameter of the target object and $q_p$ represents the ratio of the outlier pixels. The pose hypothesis with the lowest $q_p$ value among $\{q_p\}_{p=1}^{P}$ is selected as the final output pose.

Furthermore, the obtained pose can be optionally refined using the Iterative Closest Point (ICP) algorithm ICP [9,58]. The ICP refinement can be done before or after the pose hypothesis selection as shown in the experiments.

### 3.8. Combined Loss Function

The entire network consists of a single shared backbone with three head branches and is trained in an end-to-end fashion. The overall training loss is

$$L = \frac{1}{bs} \sum_{i}^{bs} (\lambda_1 \ell_i^{vp} + \lambda_2 \ell_i^{css} + \lambda_3 \ell_i^{\theta}), \tag{5}$$

where $bs$ is the batch size, and $\lambda_1, \lambda_2$ and $\lambda_3$ are weighting parameters. In our experiments, we set the ranking margins $m_\lambda^{vp} = m_\lambda^{css} = 0.1$ and the weights $\lambda_1 = 100, \lambda_2 = 10, \lambda_3 = 1$.

### 3.9. Implementation Details

We implement the method using the PyTorch [34] framework and utilize Adam solver [26] with the cosine annealing learning rate starting from $1 \times 10^{-3}$ to $1 \times 10^{-5}$ and weight decay $1 \times 10^{-5}$ for training 50 epochs (around three days) on a single Nvidia RTX3090 GPU.

**Training Data** Our training data is generated from the public 3D shape dataset [4]. Following LatentFusion [32], we exclude large objects for efficient data loading and obtain ∼19k shapes over the original 52,274 shapes. For each object, we first randomly sample 16 anchor viewpoints $\{R_i\}_{i=1}^{16}$ distributed on a sphere centered on the object. Next, we separately apply a random in-plane rotation $R_i^{\theta}$ ($R_i^{\theta} R_i$) and a random out-of-plane rotation $R_i^{\gamma}$ ($R_i^{\gamma} R_i$) for each anchor viewpoint, which results in a batch of viewpoint triplets for a single object. We randomly select eight objects each time and form a training batch with the size of 128. The Pyrender [31] library is employed to synthesize the corresponding depth images from these sampled viewpoints. Similar to [32], we use data augmentation techniques to improve the generalization of the network.

More details are provided in the supplementary material.

| Generalization | Train Data | Method | Input | ICP | VSD (%) |
|---|---|---|---|---|---|
| Single Trained Object | Real (+Syn.) | Pix2Pose [33] | RGB | | 29.5 |
| | | PVNet [36] | RGB | | 40.4 |
| | | PPFNet [11] | D | | 49.0 |
| | | PointNet++ [37] | D | | 54.0 |
| | | StablePose [39] | D | | **73.0** |
| | Syn. Only | AAE [43] | RGB | | 19.3 |
| | | AAE [43] | RGBD | ✓ | 68.6 |
| Multi-Trained Objects | Real (+Syn.) | CosyPose [27] | RGB | | 63.8 |
| | | DenseFusion [49] | RGBD | | 10.0 |
| | | Kehl-16 [25] | RGBD | ✓ | 24.6 |
| | Syn. Only | MP-Encoder [41] | RGB | | 20.5 |
| | | MP-Encoder [41] | RGBD | ✓ | **69.5** |
| Universal Objects | — | DrostPPF [12] | D | | 57.0 |
| | | VidalPPF [48] | D | | 72.0 |
| | Syn. Only | LatentFusion [32] | RGBD | | – |
| | | OVE6D(GT) | D | | *85.1* |
| | | OVE6D(GT) | D | ✓ | *89.0* |
| | | OVE6D(GT)‡ | D | ✓ | ***91.0*** |
| | | OVE6D(MRCNN) | D | | 69.4 |
| | | OVE6D(MRCNN) | D | ✓ | 73.1 |
| | | OVE6D(MRCNN)‡ | D | ✓ | **74.8** |

Table 1. Evaluation on **T-LESS**. We report the average VSD recall. ‡ represents the ICP refinement performed for all pose proposals before selection. We highlight the best performance in bold for each group. MRCNN and GT indicate using the masks provided by Mask-RCNN and the ground truth, respectively.

## 4. Experiments

**Datasets** OVE6D is evaluated on three public benchmark datasets: LINEMOD [18], Occluded LINEMOD [1], and T-LESS [21]. LINEMOD (LM) is one of the most popular datasets for single object 6D pose estimation, and it contains RGB-D images and 3D object models of 13 textureless household objects in cluttered scenes. We construct the test set following the previous works [44, 49]. We note that the training set of LINEMOD is completely ignored as OVE6D is fully trained using ShapeNet. Occluded LINEMOD (LMO) is a subset of LINEMOD for multi-object 6D pose estimation and contains eight annotated objects in 1214 testing images with heavy occlusions. T-LESS is a challenging dataset including 30 texture-less and symmetric industrial objects with highly similar shapes. The evaluation is performed on the PrimeSense test set, and we report the results for a single object per class following the protocol specified in the BOP challenge [22].

| General-ization | Train Data | Method | Input | ICP | ADD (-S)(%) |
|---|---|---|---|---|---|
| Single Trained Object | Real (+Syn.) | Self6D [50] | RGBD | | 86.9 |
| | | G2LNet [6] | RGBD | | 98.7 |
| | | PVN3D [17] | RGBD | | 99.4 |
| | | FFB6D [16] | RGBD | | **99.7** |
| | Syn. Only | Self6D [50] | RGBD | | 40.1 |
| | | AAE [43] | RGBD | ✓ | 71.6 |
| | | SSD6D [24] | RGBD | ✓ | 90.9 |
| Multi-Trained Objects | Real (+Syn.) | DenseFusion [49] | RGBD | | 94.3 |
| | | PR-GCN [59] | RGBD | | **99.6** |
| | | CloudAAE(GT) [13] | D | | *86.8* |
| | | CloudAAE(GT) [13] | D | ✓ | *95.5* |
| | Syn. Only | CloudPose (GT) [14] | D | ✓ | *75.2* |
| | | CloudAAE(GT) [13] | D | | *82.1* |
| | | CloudAAE(GT) [13] | D | ✓ | *92.5* |
| Universal Objects | Syn. Only | LatentFusion(GT) [32] | RGBD | | *87.1* |
| | | OVE6D(GT) | D | | *96.4* |
| | | OVE6D(GT) | D | ✓ | *98.3* |
| | | OVE6D(GT)‡ | D | ✓ | ***98.7*** |
| | | OVE6D(MRCNN) | D | | 86.1 |
| | | OVE6D(MRCNN) | D | ✓ | 91.4 |
| | | OVE6D(MRCNN)‡ | D | ✓ | **92.4** |

Table 2. Evaluation on **LINEMOD**. We report the average ADD(-S) recall. We highlight the best performance in bold for each group. ‡ represents ICP refinement performed for all pose proposals before selection. MRCNN and GT indicate using the masks provided by Mask-RCNN and the ground truth, respectively.

| General-ization | Train Data | Method | Input | ICP | ADD (-S)(%) |
|---|---|---|---|---|---|
| Single Trained Object | Real (+Syn.) | PVNet [36] | RGB | | 42.4 |
| | | PVN3D [17] | RGBD | | 63.2 |
| | | FFB6D [16] | RGBD | | 66.2 |
| | | PVNet [36] | RGBD | ✓ | **79.0** |
| Multi-Trained Objects | Real +Syn. | PoseCNN [54] | RGB | | 24.9 |
| | | PR-GCN [59] | RGBD | | 65.0 |
| | | PoseCNN [54] | RGBD | ✓ | **78.0** |
| | | CloudAAE(GT) [13] | D | | *58.9* |
| | | CloudAAE(GT) [13] | D | ✓ | *66.1* |
| | Syn. Only | CloudPose (GT) [14] | D | ✓ | *44.2* |
| | | CloudAAE(GT) [13] | D | | *57.1* |
| | | CloudAAE(GT) [13] | D | ✓ | *63.2* |
| Universal Objects | Syn. Only | LatentFusion [32] | RGBD | | - |
| | | OVE6D(GT) | D | | *70.9* |
| | | OVE6D(GT) | D | ✓ | *80.0* |
| | | OVE6D(GT)‡ | D | ✓ | ***82.5*** |
| | | OVE6D(MRCNN) | D | | 56.1 |
| | | OVE6D(MRCNN) | D | ✓ | 70.3 |
| | | OVE6D(MRCNN)‡ | D | ✓ | **72.8** |

Table 3. Evaluation on **Occluded LINEMOD**. We report the average ADD(-S) and VSD recalls. ‡ depicts ICP refinement performed for all pose proposals before selection. We highlight the best performance in bold for each group. MRCNN and GT indicate using the masks provided by Mask-RCNN and the ground truth, respectively.

**Segmentation Mask** The object segmentation mask is one of the inputs to the proposed pose estimation method. In the experiments, we obtain the masks using off-the-shelf implementation of Mask-RCNN [15] from the Detectron2 [53] library. We train Mask-RCNN using a large set of synthetic images generated from the object models. We use the class labels provided by Mask-RCNN as object IDs in the experiments. In addition, we also report the results for ground truth segmentation masks.

**Metrics and Configurations** We follow prior works [19, 41, 42] and report the results in terms of two standard 6D pose estimation metrics ADD(-S) [19, 22] (for LM and LMO) and VSD [22] (for T-LESS). Please refer to [19, 22] for more details. Furthermore, we use $N = 4000$, $K = 50$, and $P = 5$ for OVE6D if not otherwise stated.

### 4.1. Comparison with the state-of-the-art

We compare OVE6D against the recent (mainly learning-based) pose estimation works using the popular T-LESS, LINEMOD, and Occluded LINEMOD datasets. We categorize the methods into three main groups in terms of generalization. The methods in the first and second groups train a separate model for each individual object or a model for multiple objects, respectively. The third group consists of methods that do not require any dataset-specific training, other than obtaining the 3D models of the target objects. We further split these groups into approaches that use real and/or synthetic data during model training. Notably, OVE6D and LatentFusion belong to the third category and use only synthetic data for training.

**T-LESS** The results for OVE6D and the baseline methods are reported in Table 1 in terms of VSD metric. We do not report the results for LatentFusion as it does not perform well due to the occlusions. Note that all other learning-based methods are trained on the T-LESS dataset, unlike OVE6D, which is trained on ShapeNet. Nevertheless, OVE6D still achieves the state-of-the-art performance. In particular, OVE6D with ICP improves over the recent state-of-the-art method StablePose [39] by noticeable margin of 1.8%, regardless that StablePose trains a separate model for each object using real-world examples with pose annotations. The non-learning based VidalPPF [48] also performs well, but the approach is computationally expensive. Moreover, OVE6D achieves 91% recall when using the ground truth masks, indicating the potential for performance improvement with better segmentation masks. The results indicate that OVE6D is particularly suitable for texture-less and symmetric industrial objects.

**LINEMOD and Occluded LINEMOD** The results for the LINEMOD (LM) and Occluded LINEMOD (LMO) datasets are reported in tables 2 and 3, respectively. All methods, except for OVE6D and LatentFusion [32], are specifically trained for the LM dataset. The LatentFusion

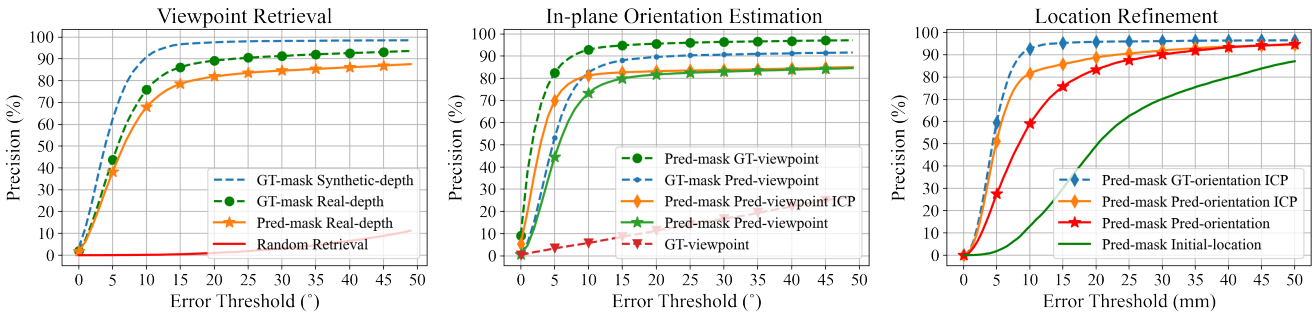



Figure 6. **Evaluation of OVE6D submodules**. The precision values for different error thresholds for the viewpoint retrieval (left), in-plane orientation regression (middle), and location refinement (right) modules using the LINEMOD dataset. "Real-depth" and "Synthetic-depth" separately refer to real-world depth images and the synthesized depth images (using the ground truth poses). "GT-*" and "Pred-*" indicate using the corresponding ground truth and predicted results of *, respectively.

results are reported only for LM due to heavy occlusions in LMO. In general, the RGBD based methods, trained with real-world and synthetic data, achieve the best performance (*e.g.* 99.7% recall on LM with FFB6D [16] and 79.0% recall on LMO with PVNet [36]). However, OVE6D obtains competitive results, particularly when compared to the methods trained with purely synthetic data. Without ICP refinement, OVE6D obtains 86.1% and 56.1% recall for LM and LMO, respectively. In addition, OVE6D with ICP results in 73% recall on LMO, which shows that OVE6D is able to generalize to real-world scenes, even in the case of heavy clutter and occlusion. Moreover, compared with LatentFusion [32], another universal model trained on ShapeNet, OVE6D obtains better results with a clear margin of 9.3% (96.4% vs. 87.1%) on LM, while relying only depth information in the pose estimation.

### 4.2. Additional Experiments

**Parameter Configuration** The main parameters in OVE6D are the sampling number of viewpoints (N), the retrieving number of viewpoint candidates (K), and the number of orientation proposals (P). We examined how these parameter values affect the performance, and observed that the method is relatively stable over a wide range of different settings. We found $N = 4000$, $K = 50$, and $P = 5$ to be a good trade-off between the accuracy and the efficiency. The detailed results are provided in the supplementary material.

**Viewpoint Retrieval** Figure 6 (left) illustrates the performance of the viewpoint retrieval module over multiple thresholds using estimated and ground-truth segmentation masks on the LINEMOD dataset. In this experiment, we consider only a single, top scoring, pose hypothesis. We note that already 70% of the cases are retrieved with higher than 10° accuracy. In addition, the gap between synthetic and real data is relatively small, indicating good generalization despite of the domain gap.

**In-plane Orientation Estimation** The results for in-plane rotation module are illustrated in Figure 6 (middle). Given viewpoints, retrieved with the predicted masks, we reach up to 73% precision at the 10° error threshold with a single forward pass, and further improve it to 80% with ICP refinement. We also note that by using the ground truth viewpoint, the precision can be further improved over 90% even without ICP.

**Location Refinement** Figure 6 (right) illustrates the performance of the proposed non-parametric location refinement module. We observe that, at the error tolerance of 10mm, the refinement can improve the precision from the initial estimate 13% to 60% and further to 81% with the ICP refinement. Thus, the proposed refinement module clearly improves the translation estimation with or without ICP refinement.

**Inference Time** The full pose inference with OVE6D requires approximately 50ms per object with Nvidia RTX3090 GPU and AMD 835 Ryzen 3970X CPU. In comparison, LatentFusion requires roughly 20 seconds per object with 100 back propagation iterations.

## 5. Discussion, Limitations and Conclusion

In this work, we proposed a model called OVE6D for inferring the object 6D pose in a cascaded fashion. The model was trained using a large body of synthetic 3D objects and assessed using three challenging real-world benchmark datasets. The results demonstrate that the model generalizes well to unseen data without needing any parameter optimization, which significantly simplifies the addition of novel objects and enables use cases with thousands of objects. The main limitations of this approach include the requirements for the object 3D mesh model and instance segmentation mask, which may not always be easy to obtain.

## References


[1] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6d object pose estimation using 3d object coordinates. In *European conference on computer vision*, pages 536–551. Springer, 2014. 2, 6

[2] Mai Bui, Sergey Zakharov, Shadi Albarqouni, Slobodan Ilic, and Nassir Navab. When regression meets manifold learning for object recognition and pose estimation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6140–6146. IEEE, 2018. 3

[3] Yannick Bukschat and Marcus Vetter. Efficientpose: An efficient, accurate and scalable end-to-end 6d multi object pose estimation approach, 2020. 2

[4] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 2, 3, 4, 6

[5] Dengsheng Chen, Jun Li, Zheng Wang, and Kai Xu. Learning canonical shape space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11973–11982, 2020. 3

[6] Wei Chen, Xi Jia, Hyung Jin Chang, Jinming Duan, and Ales Leonardis. G2l-net: Global to local network for real-time 6d pose estimation with embedding vector features. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4233–4242, 2020. 1, 3, 7

[7] Wei Chen, Xi Jia, Hyung Jin Chang, Jinming Duan, Linlin Shen, and Ales Leonardis. Fs-net: Fast shape-based network for category-level 6d object pose estimation with decoupled rotation mechanism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1581–1590, 2021. 3

[8] Xu Chen, Zijian Dong, Jie Song, Andreas Geiger, and Otmar Hilliges. Category level object pose estimation via neural analysis-by-synthesis. In *European Conference on Computer Vision*, pages 139–156. Springer, 2020. 3

[9] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992. 6

[10] 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

[11] 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. 6

[12] Bertram Drost, Markus Ulrich, Nassir Navab, and Slobodan Ilic. Model globally, match locally: Efficient and robust 3d object recognition. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 998–1005. Ieee, 2010. 6

[13] Ge Gao, Mikko Lauri, Xiaolin Hu, Jianwei Zhang, and Simone Frintrop. Cloudaae: Learning 6d object pose regression with on-line data synthesis on point clouds. *arXiv preprint arXiv:2103.01977*, 2021. 3, 7

[14] 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. 3, 7

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 7

[16] Yisheng He, Haibin Huang, Haoqiang Fan, Qifeng Chen, and Jian Sun. Ffb6d: A full flow bidirectional fusion network for 6d pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3003–3013, 2021. 1, 3, 7, 8

[17] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 3, 7

[18] Stefan Hinterstoisser, Stefan Holzer, Cedric Cagniart, Slobodan Ilic, Kurt Konolige, Nassir Navab, and Vincent Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *2011 international conference on computer vision*, pages 858–865. IEEE, 2011. 1, 6

[19] 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. 3, 7

[20] 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. 2

[21] Tomáš Hodaň, Pavel Haluza, Štěpán Obdržálek, Jiří Matas, Manolis Lourakis, and Xenophon Zabulis. T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017. 2, 6

[22] Tomas Hodan, Frank Michel, Eric Brachmann, Wadim Kehl, Anders GlentBuch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, et al. Bop: Benchmark for 6d object pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–34, 2018. 1, 6, 7

[23] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28:2017–2025, 2015. 5

[24] 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 *Proceedings of*

*the IEEE international conference on computer vision*, pages 1521–1529, 2017. 1, 2, 7

[25] Wadim Kehl, Fausto Milletari, Federico Tombari, Slobodan Ilic, and Nassir Navab. Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation. In *European conference on computer vision*, pages 205–220. Springer, 2016. 3, 6

[26] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[27] 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. 6

[28] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155, 2009. 2

[29] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 683–698, 2018. 3

[30] 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

[31] Matthew Matl. Pyrender. https://github.com/mmatl/pyrender, 2019. 6

[32] Keunhong Park, Arsalan Mousavian, Yu Xiang, and Dieter Fox. Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 1, 3, 4, 6, 7, 8

[33] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019. 2, 6

[34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019. 6

[35] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2011–2018. IEEE, 2017. 2

[36] Sida Peng, Xiaowei Zhou, Yuan Liu, Haotong Lin, Qixing Huang, and Hujun Bao. Pvnet: pixel-wise voting network for 6dof object pose estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 1, 2, 6, 7, 8

[37] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 6

[38] 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

[39] 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. 1, 3, 6, 7

[40] 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 *2020 International Conference on 3D Vision (3DV)*, pages 291–300. IEEE, 2020. 2

[41] Martin Sundermeyer, Maximilian Durner, En Yen Puang, Zoltan-Csaba Marton, Narunas Vaskevicius, Kai O Arras, and Rudolph Triebel. Multi-path learning for object pose estimation across domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13916–13925, 2020. 1, 2, 3, 6, 7

[42] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *The European Conference on Computer Vision (ECCV)*, September 2018. 2, 3, 7

[43] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, and Rudolph Triebel. Augmented autoencoders: Implicit 3d orientation learning for 6d object detection. *International Journal of Computer Vision*, 128(3):714–729, 2020. 1, 3, 6, 7

[44] Bugra Tekin, Sudipta N Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 292–301, 2018. 2, 6

[45] Meng Tian, Marcelo H Ang, and Gim Hee Lee. Shape prior deformation for categorical 6d object pose and size estimation. In *European Conference on Computer Vision*, pages 530–546. Springer, 2020. 3

[46] Ameni Trabelsi, Mohamed Chaabane, Nathaniel Blanchard, and Ross Beveridge. A pose proposal and refinement network for better 6d object pose estimation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2382–2391, 2021. 2

[47] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 2

[48] Joel Vidal, Chyi-Yeu Lin, Xavier Lladó, and Robert Martí. A method for 6d pose estimation of free-form rigid objects using point pair features on range data. *Sensors (Basel, Switzerland)*, 18, 2018. 6, 7

[49] 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/CVF conference on computer vision and pattern recognition*, pages 3343–3352, 2019. 1, 3, 6, 7

[50] Gu Wang, Fabian Manhardt, Jianzhun Shao, Xiangyang Ji, Nassir Navab, and Federico Tombari. Self6d: Self-supervised monocular 6d object pose estimation. In *European Conference on Computer Vision*, pages 108–125. Springer, 2020. 2, 7

[51] 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

[52] Paul Wohlhart and Vincent Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3109–3118, 2015. 3

[53] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 7

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

[55] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. Pointfusion: Deep sensor fusion for 3d bounding box estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 244–253, 2018. 3

[56] Sergey Zakharov, Wadim Kehl, Benjamin Planche, Andreas Hutter, and Slobodan Ilic. 3d object instance recognition and pose estimation using triplet loss with dynamic margin. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 552–559. IEEE, 2017. 3

[57] 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. 2

[58] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International journal of computer vision*, 13(2):119–152, 1994. 6

[59] Guangyuan Zhou, Huiqun Wang, Jiaxin Chen, and Di Huang. Pr-gcn: A deep graph convolutional network with point refinement for 6d pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2793–2802, October 2021. 1, 7