

Fusing Local Similarities for Retrieval-based 3D Orientation Estimation of Unseen Objects

Chen Zhao, Yinlin Hu, and Mathieu Salzmann

CVLab, EPFL, Switzerland

{chen.zhao,yinlin.hu, mathieu.salzmann}@epfl.ch

Abstract. In this paper, we tackle the task of estimating the 3D orientation of previously-unseen objects from monocular images. This task contrasts with the one considered by most existing deep learning methods which typically assume that the testing objects have been observed during training. To handle the unseen objects, we follow a retrieval-based strategy and prevent the network from learning object-specific features by computing multi-scale local similarities between the query image and synthetically-generated reference images. We then introduce an adaptive fusion module that robustly aggregates the local similarities into a global similarity score of pairwise images. Furthermore, we speed up the retrieval process by developing a fast clustering-based retrieval strategy. Our experiments on the LineMOD, LineMOD-Occluded, and T-LESS datasets show that our method yields a significantly better generalization to unseen objects than previous works.

Keywords: Object 3D Orientation Estimation, Unseen Objects

1 Introduction

Estimating the 3D orientation of objects from an image is pivotal to many computer vision and robotics tasks, such as robotic manipulation [7,41,30], augmented reality, and autonomous driving [11,6,39,23]. Motivated by the tremendous success of deep learning, much effort [37,25,33] has been dedicated to developing deep networks able to recognize the objects depicted in the input image and estimate their 3D orientation. To achieve this, most learning-based methods assume that the training data and testing data contain exactly the same objects [16,34] or similar objects from the same category [35,22]. However, this assumption is often violated in real-world applications, such as robotic manipulation, where one would typically like the robotic arm to be able to handle previously-unseen objects without having to re-train the network for them.

In this paper, as illustrated in Fig. 1, we tackle the task of 3D orientation estimation for *previously-unseen* objects. Specifically, we develop a deep network that can be trained on a limited number of objects, and yet remains effective when tested on novel objects that drastically differ from the training ones in terms of both appearance and shape. To handle such previously-unseen objects, we cast the task of 3D orientation estimation as an image retrieval problem.



Fig. 1. 3D Orientation Estimation for Unseen Objects. The network is trained on a limited number of objects and tested on unseen (new) objects that fundamentally differ from the training ones in shape and appearance. The goal is to predict both the category and the 3D orientation of these unseen objects.

We first create a database of synthetic images depicting objects in different orientation. Then, given a real query image of an object, we search for the most similar reference image in the database, which thus indicates both the category and 3D orientation of this object.

Intuitively, image retrieval methods [36] offer a promising potential for generalization, because they learn the relative similarity of pairwise images, which can be determined without being aware of the object category. However, most previous works [36,29,1,31] that follow this approach exploit a global image representation to measure image similarity, ignoring the risk that a global descriptor may integrate high-level semantic information coupled with the object category, which could affect the generalization ability to unseen objects. To address this problem, our approach relies on the similarities of local patterns, which are independent to the object category and then facilitate the generalization to new objects. Specifically, we follow a multi-scale strategy and extract feature maps of different sizes from the input image. To facilitate the image comparison process, we then introduce a similarity fusion module, adaptively aggregating multiple local similarity scores into a single one that represents the similarity between two images. To further account for the computational complexity of the resulting multi-scale local comparisons, we design a fast clustering-based retrieval strategy.

We conduct experiments on three public datasets, LineMOD [13], LineMOD-Occluded (LineMOD-O) [3], and T-LESS [14], comparing our method with both hand-crafted [8] and deep learning [36,1,38,28] approaches. Our empirical results evidence the superior generalization ability of our method to previously-unseen objects. Furthermore, we perform ablation studies to shed more light on the effectiveness of each component in our method¹. Our contributions can be summarized as follows:

- We estimate the 3D orientation of previously-unseen objects by introducing an image retrieval framework based on multi-scale local similarities.
- We develop a similarity fusion module, robustly predicting an image similarity score from multi-scale pairwise feature maps.

¹ The code will be publicly released.

- We design a fast clustering-based retrieval strategy that achieves a good trade-off between the 3D orientation estimation accuracy and efficiency.

2 Related Work

Object Pose Estimation. In recent years, deep learning has been dominating the field of object pose estimation. For instance, PoseCNN [37] relies on two branches to directly predict the object orientation as a quaternion, and the 2D location of the object center, respectively. PVNet [25] estimates the 2D projections of 3D points using a voting network. The object pose is then recovered by using a PnP algorithm [10] over the predict 2D-3D correspondences. DenseFusion [33] fuses 2D and 3D features extracted from RGB-D data, from which it predicts the object pose. GDR-Net [34] predicts a dense correspondence map, acting as input to a Patch-PnP module that recovers the object pose. These deep learning methods have achieved outstanding pose estimation accuracy when the training and testing data contain the same object instances [9]. However, the patterns they learn from the input images are instance specific, and these methods cannot generalize to unseen objects [24].

Category-Level Object Pose Estimation. Some methods nonetheless loosen the constraint of observing the same object instances at training and testing time by performing category-level object pose estimation [35,5,32,18]. These methods assume that the training data contain instances belonging to a set of categories, and new instances from these categories are observed during testing. In this context, a normalized object coordinate space (NOCS) is typically used [35,18], providing a canonical representation shared by different instances within the same category. The object pose is obtained by combining the NOCS maps, instance masks, and depth values. These methods rely on the intuition that the shapes of different instances in the same category are similar, and then the patterns learned from the training data can generalize to new instances in the testing phase. As such, these methods still struggle in the presence of testing objects from entirely new categories. Furthermore, all of these techniques require *depth* information as input. By contrast, our method relies only on RGB images, and yet can handle unseen objects from new categories at testing time.

Unseen Object Pose Estimation. A few attempts at predicting the pose of unseen objects have been made in the literature. In particular, LatentFusion [24] introduces a latent 3D representation and optimizes an object’s pose by differentiable rendering. DeepIM [19] presents an iterative framework, using a matching network to optimize an initial object pose. Both of these methods require an *initial* pose estimate, which is typically hard to be obtained for unseen objects. Furthermore, the pose estimation step in LatentFusion leverages *depth* information. Since estimating the full 6D pose of an *unseen* object from a single RGB image is highly challenging, several works suggest simplifying this problem by focusing on estimating the 3D object orientation [36,2,29,38]. These methods

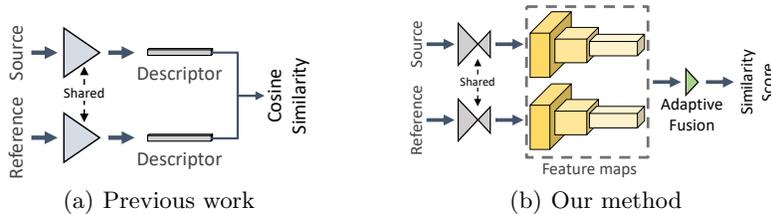


Fig. 2. Difference between previous works and our method. (a) Existing works convert images into global descriptors that are used to compute a similarity score for retrieval. (b) Our method compares images using local similarities between the corresponding elements in feature maps, and adaptively fuses these local similarities into a single one that indicates the similarity of two images.

utilize the 3D object model to generate multi-view references, which are then combined with the real image either to perform template matching [36,2,29,28] or to directly regress the 3D orientation [38]. However, they propose to learn a global representation from an image, in which the high-level semantic information is correlated to the object and then limits the generalization to unseen objects. In this paper, we also focus on 3D orientation estimation of unseen objects, but handle this problem via a multi-scale local similarity learning network.

3 Method

3.1 Problem Formulation

Let us assume to be given a set of training objects \mathcal{O}_{train} belonging to different categories \mathcal{C}_{train} .² As depicted by Fig. 1, we aim to train a model that can predict the 3D orientation of new objects \mathcal{O}_{test} , $\mathcal{O}_{test} \cap \mathcal{O}_{train} = \emptyset$, from entirely new categories \mathcal{C}_{test} , $\mathcal{C}_{test} \cap \mathcal{C}_{train} = \emptyset$. Specifically, given an RGB image \mathbf{I}_{src} containing an object $O_{src} \in \mathcal{O}_{test}$, our goal is to both recognize the object category C_{src} and estimate the object’s 3D orientation, expressed as a rotation matrix $\mathbf{R}_{src} \in \mathbb{R}^{3 \times 3}$. We tackle this dual problem as an image retrieval task. For each $O_{src}^i \in (\mathcal{O}_{train} \cup \mathcal{O}_{test})$, we generate references \mathcal{I}_{ref}^i with different 3D orientations by rendering the corresponding 3D model \mathbf{M}_i . We then seek to pick $\hat{\mathbf{I}}_{ref} \in \{\mathcal{I}_{ref}^1 \cup \mathcal{I}_{ref}^2 \dots \cup \mathcal{I}_{ref}^N\}$ that is the most similar to \mathbf{I}_{src} . The category label C_{src} and 3D orientation \mathbf{R}_{src} of O_{src} are then taken as those of the corresponding \hat{O}_{ref} .

3.2 Motivation

As illustrated in Fig. 2(a), the existing retrieval-based 3D orientation estimation methods [36,29] convert an image to a global descriptor. Retrieval is then

² In our scenario, and in contrast to category-level pose estimation, each object instance corresponds to its own category.

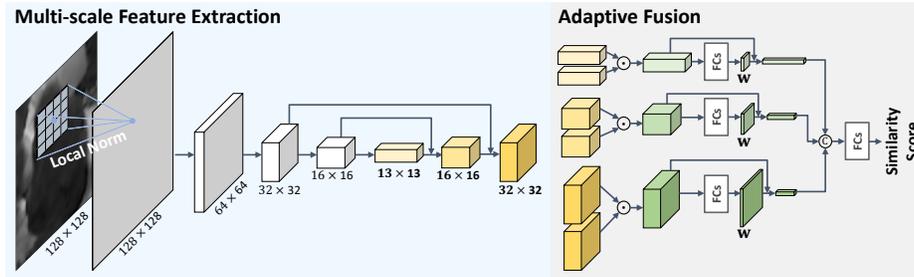


Fig. 3. Network architecture. We extract multi-scale features from a locally-normalized image. We then compute local similarities at each scale between the features of the source image and those of a reference one, and adaptively fuse them into a global similarity score.

performed by computing the similarity between pairs of descriptors. As a consequence, the deep network that extracts the global descriptor typically learns to encode object-specific semantic information in the descriptor, which results in a limited generalization ability to unseen objects. By contrast, we propose to compare images via local patch descriptors, in which it is harder to encode high-level semantic information thus encouraging the network to focus on local geometric attributes. As shown in Fig. 2(b), we estimate local similarities between the corresponding elements in source and reference feature maps. Furthermore, to enforce robustness to noise, such as background, we introduce an adaptive fusion module capable of robustly predicting an image similarity score from the local ones.

3.3 Multi-scale Patch-level Image Comparison

As the source image is real but the reference ones are synthetic with discretely sampled 3D orientation, the appearance and shape variations are inevitable even for the most similar reference. Moreover, the background included in the source image, but absent from the reference ones, typically interferes with our patch-level comparisons. In practice, we have observed that small patches could be too sensitive to appearance and shape variations, while large patches tend to be affected by the background. Finding a single effective patch size balancing robustness to the domain gap and to the background therefore is challenging.

To address this issue, we introduce a multi-scale feature extraction module. As shown in Fig. 3, our network takes a grey-scale image $\mathbf{I} \in \mathbb{R}^{128 \times 128}$ as input, which shows better robustness than color images in practice. Subsequently, we employ a series of ResNet layers [12], estimating a down-sampled feature map $\mathbf{F} \in \mathbb{R}^{13 \times 13 \times C}$. We compute multi-scale feature representations by progressively up-sampling \mathbf{F} using deconvolution layers [20] and bilinear interpolation. We also utilize skip connections [27] to better preserve the geometric information. The elements in the generated multi-scale feature maps then encode patches of different sizes in \mathbf{I} , which enables multi-scale patch-level image comparison.

To perform image retrieval, one nonetheless needs to compute a single similarity score for a pair of images. To this end, we compare the pairwise multi-scale feature maps and fuse the resulting local similarities into a single score expressed as

$$s = f(g(\mathbf{F}_{src}^1, \mathbf{F}_{ref}^1), g(\mathbf{F}_{src}^2, \mathbf{F}_{ref}^2), \dots, g(\mathbf{F}_{src}^S, \mathbf{F}_{ref}^S)), \quad (1)$$

where \mathbf{F}_{src} and \mathbf{F}_{ref} represent the feature maps of \mathbf{I}_{src} and \mathbf{I}_{ref} , respectively, and S denotes the number of scales. A straightforward solution to estimate s is to compute the per-element cosine similarity for all pairs $(\mathbf{F}_{src}^i, \mathbf{F}_{ref}^i)$, with $i \in \{1, 2, \dots, S\}$, and average the resulting local similarities. However, this strategy would not be robust to outlier patches, such as those dominated by background content. Therefore, we introduce an adaptive fusion strategy illustrated in the right part of Fig. 3. Following the same formalism as above, it computes an image similarity score as

$$s = f(\text{cat}[g(\mathbf{F}_i^* \odot \mathbf{w}_i)], \psi), i \in \{1, 2, \dots, S\}, \quad (2)$$

where cat indicates the concatenation process, $g: \mathbb{R}^{H \times W \times C} \rightarrow \mathbb{R}^C$ denotes the summation over the spatial dimensions, \mathbf{F}_i^* represents the local similarities obtained by computing the cosine similarities between the corresponding elements in \mathbf{F}_{src}^i and \mathbf{F}_{ref}^i , \odot indicates the Hadamard product, and ψ represents the learnable parameters of the fully connected layers (FCs) $f(\cdot)$. The weights \mathbf{w}_i encode a confidence map over \mathbf{F}_i^* to account for outliers, and are computed as

$$\mathbf{w}_i = \frac{\exp(h(\mathbf{F}_i^*, \omega)) \odot \text{sigmoid}(q(\mathbf{F}_i^*, \theta))}{\sum \exp(h(\mathbf{F}_i^*, \omega)) \odot \text{sigmoid}(q(\mathbf{F}_i^*, \theta))}, \quad (3)$$

where ω and θ are learnable parameters of the fully connected layers $h(\cdot)$ and $q(\cdot)$, respectively. This formulation accounts for both the individual confidence of each element in \mathbf{F}_i^* via the sigmoid function, and the relative confidence w.r.t. all elements jointly via the softmax function. As such, it models both the local and global context of \mathbf{F}_i^* , aiming to decrease the confidence of the outliers while increases that of the inliers. Our experimental results in Section 4.5 show that our adaptive fusion yields better results than the straightforward averaging process described above, even when trained in an unsupervised manner.

To further reduce the effects of object-related patterns in local regions and synthetic-to-real domain gap, we pre-process \mathbf{I} via a local normalization. Each pixel \bar{p}_{ij} in the normalized image is computed as

$$\bar{p}_{ij} = \frac{p_{ij} - \mu}{\sigma}, \quad (4)$$

where p_{ij} is the corresponding pixel in \mathbf{I} , and

$$\mu = \frac{1}{r^2} \sum_{i', j'} p_{i' j'}, \quad \sigma = \sqrt{\frac{1}{r^2} \sum_{i', j'} (p_{i' j'} - \mu)^2}, \quad (5)$$

with $i' \in [i \pm r/2]$, $j' \in [j \pm r/2]$, and r denoting the window size.

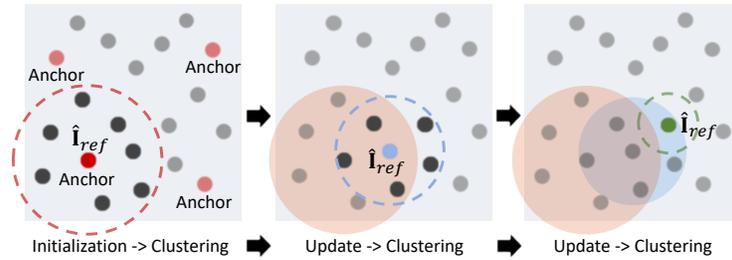


Fig. 4. Fast clustering-based retrieval. The location of the source image in the reference database is initialized by comparing the source image with a set of anchors. $\hat{\mathbf{I}}_{ref}$ is dynamically updated after each clustering step until convergence, generating the final retrieved result.

3.4 Fast Clustering-based Retrieval

Although the proposed patch-level image comparison integrates more local geometric information than the image-level methods [36,29], and as will be shown by our experiments thus yields better generalization to unseen objects, it suffers from a high retrieval time. Indeed, a naïve image retrieval strategy compares \mathbf{I}_{src} with every reference in the database. Given N objects with R reference images each, the cost of $O(NR)$ quickly becomes unaffordable as N and R increase. This could be remedied by parallel computing, but at the cost of increasing memory consumption. Here, we therefore introduce a fast clustering-based retrieval method that balances effectiveness and efficiency.

As illustrated in Fig. 4, instead of comparing \mathbf{I}_{src} with all the references one-by-one, we first roughly locate \mathbf{I}_{src} in the reference database and then iteratively refine this initial location. Our method is summarized in Algorithm 1. We omit some subscripts for convenience. Specifically, for each object, we first sample k_{ac} anchors from \mathcal{I}_{ref} using farthest point sampling (FPS), which leads to a good coverage [26] of \mathcal{I}_{ref} . This is done using the geodesic distance of the corresponding 3D rotation matrices as a metric in FPS. The anchor with the largest score s computed from Eq. 2 is taken as the initial point $\hat{\mathbf{I}}_{ref}$. Subsequently, we perform retrieval using the method described in Section 3.3 within a local region centered at the current $\hat{\mathbf{I}}_{ref}$, and update $\hat{\mathbf{I}}_{ref}$ based on the similarity scores. Such updates are performed until convergence.

The straightforward application of this strategy would be prone to local optima. Intuitively, this can be addressed by increasing the size of the local region, but this would come at a higher computational cost. Therefore, we determine a search space and further make use of FPS to select anchors within the space, because FPS covers a larger region than KNN with the same number of samples. At each iteration, we decrease the radius of the search space to further improve efficiency.

Algorithm 1: Fast Clustering-based Retrieval

Input: $\mathbf{I}_{src}, \mathcal{I}_{ref}, \mathcal{R}_{ref} = \{\mathbf{R}_1, \mathbf{R}_2, \dots, \mathbf{R}_R\}, k_{ac}, R$
Output: $\hat{\mathbf{I}}_{ref}, \mathbf{R}_{src}$

- 1 Sample k_{ac} anchors from \mathcal{I}_{ref} using FPS;
- 2 Estimate similarities using Eq. 2;
- 3 Initialize $\hat{\mathbf{I}}_{ref}$ as the most similar anchor;
- 4 $j = 1$;
- 5 **repeat**
- 6 Define a search space around $\hat{\mathbf{I}}_{ref}$ with a radius of $\lfloor R/2^j \rfloor$;
- 7 Compute anchors using FPS;
- 8 Estimate similarities using Eq. 2;
- 9 Update $\hat{\mathbf{I}}_{ref}$;
- 10 $j++$;
- 11 **until** $\hat{\mathbf{I}}_{ref}$ converges;
- 12 Determine \mathbf{R}_{src} as $\hat{\mathbf{R}}_{ref} \in \mathcal{R}_{ref}$.

3.5 Training and Testing

In the training stage, we follow the infoNCE contrastive learning formalism [4]. We associate each training sample (source image) in a mini-batch with its closest reference to form a positive pair. To better cover the entire training set, we also group each positive pair with a random sample, leading to a triplet. Note that in the standard infoNCE loss [4] for contrastive learning, all samples except for the most similar one are treated as negative. In our context, all reference images would be penalized equally, except for the one from the same category and with the closest 3D orientation. Their 3D orientation difference is thus not differentiated. To better account for the continuous nature of the 3D orientation difference and avoid over-penalizing some references, we introduce a weighted-infoNCE loss

$$\ell_{ij} = -\log \frac{\exp(s_{ij}/\tau) \cdot w_{ij}}{\sum_{k=1}^{3B} \exp(s_{ik}/\tau) \cdot w_{ik}}, \quad (6)$$

where s_{ij} is the similarity score of the positive pair $(\mathbf{I}_i, \mathbf{I}_j)$, $\tau = 0.1$ denotes a temperature parameter [4], B is the size of a mini-batch, and w_{ij} (and similarly w_{ik}) represents a weight computed as

$$w_{ij} = \begin{cases} \arccos(\frac{\text{tr}(\mathbf{R}_i^T \mathbf{R}_j) - 1}{2})/\pi & \text{if } C_j = C_i \\ 1 & \text{else} \end{cases}. \quad (7)$$

In the testing phase, we store the features of the reference images and sample the initial anchors offline; we recognize the object categories and predict the 3D orientation online. More specifically, we first compare \mathbf{I}_{src} with k_{ac} anchors for each \mathcal{I}_{ref}^i , and take C_{src} to be the category of the anchor with the largest similarity score. This process reduces the complexity of object category recognition from $O(NR)$ to $O(Nk_{ac})$. Subsequently, we restrict the search for $\hat{\mathbf{I}}_{ref}$ in

our clustering-based retrieval process to the references depicting the recognized object. \mathbf{R}_{src} is finally taken as the corresponding rotation matrix $\hat{\mathbf{R}}_{ref}$.

4 Experiments

4.1 Implementation Details

In our experiments, we set the window size for local normalization and the number of anchors for fast retrieval to $r = 32$ and $k_{ac} = 1024$, respectively. We train our network for 200 epochs using the Adam [17] optimizer with a batch size of 16 and a learning rate of 10^{-4} , which is divided by 10 after 50 and 150 epochs. Training takes around 20 hours on a single NVIDIA Tesla V100. Please refer to the supplementary material for the detail of our network architecture.

4.2 Experimental Setup

Following the standard setting [36,29,38,28], we assume to have access to 3D object models, which provide us with canonical frames, without which the object orientation in the camera frame would be ill-defined. More explanations about the importance of a canonical frame can be found in the supplementary material. Note that the 3D object models are also used to generate the reference images. We generate $R = 10,000$ reference images for each object by rendering the 3D model with different 3D orientation. We randomly sample \mathbf{R}_{ref} using a 6D continuous representation [40], which results in better coverage of the orientation space. We compare the distributions of \mathbf{R}_{ref} sampled from the 6D continuous representation and using Euler angles [29] in the supplementary material. Following [36,24,28], we crop the objects from the source images using the provided bounding boxes.

We compare our method with both a hand-crafted approach, i.e., HOG [8], and deep learning ones, i.e., LD [36], NetVLAD [1], PFS [38], MPE [28], and GDR-Net [34]. Note that DeepIM [19] and LatentFusion [24] are not evaluated since DeepIM requires an object pose initialization and LatentFusion needs additional depth information. We also exclude [29] because it requires training a separate autoencoder for every object, and thus cannot be used to estimate orientation for an unseen object without training a new autoencoder.

4.3 Experiments on LineMOD and LineMOD-O

We first conduct experiments on LineMOD [13] and LineMOD-O [3]. We split the cropped data into three non-overlapping groups, i.e., Split #1, Split #2, and Split #3, according to the depicted objects. Please refer to the supplementary material for more details. Deep learning models are trained on LineMOD and tested on both LineMOD and LineMOD-O. We augment training data by random occlusions when the evaluation is performed on LineMOD-O. In the case of unseen objects, we select one of the three groups as the testing set. We remove

all images belonging to this group from training data, which ensures that no testing objects are observed in the training stage. In the case of seen objects on LineMOD, we separate 10% from each group of this dataset for testing. We assume the object category to be unknown during testing, and therefore employ the evaluated methods to classify the object and then predict its 3D orientation.

We thus evaluate the tested methods in terms of both object classification accuracy and 3D orientation estimation accuracy. These are computed as

$$\text{Class. Acc.} = \begin{cases} 1 & \text{if } \hat{C}_{ref} = C_{src} \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

and

$$\text{Rota. Acc.} = \begin{cases} 1 & \text{if } d(\hat{\mathbf{R}}_{ref}, \mathbf{R}_{src}) < \lambda \text{ and } \hat{C}_{ref} = C_{src} \\ 0 & \text{otherwise} \end{cases}, \quad (9)$$

respectively, with $\lambda = 30^\circ$ a predefined threshold [38] and $d(\hat{\mathbf{R}}_{ref}, \mathbf{R}_{src})$ the geodesic distance [36] between two rotation matrices. This distance is defined as

$$d(\hat{\mathbf{R}}_{ref}, \mathbf{R}_{src}) = \arccos\left(\frac{\text{tr}(\mathbf{R}_{src}^T \hat{\mathbf{R}}_{ref}) - 1}{2}\right)/\pi. \quad (10)$$

We provide the results for LineMOD and LineMOD-O in Table 1 and Table 2, respectively. As PFS assumes the object category is known, we only report its 3D orientation estimation accuracy. We replace the detection module in GDR-Net with the ground-truth bounding boxes in the presence of unseen objects since this detector cannot be used to detect unseen objects. Therefore, the classification accuracy of GDR-Net is not reported in this case. Being a traditional method, HOG does not differentiate seen and unseen objects, and thus achieves comparable results in both cases. However, its limited accuracy indicates that HOG suffers from other challenges, such as the appearance difference between real and synthetic images, and the presence of background and of occlusions. The previous retrieval-based methods, i.e, LD, NetVLAD, and MPE, achieve remarkable performance in the case of seen objects, but their accuracy significantly drops in the presence of unseen ones. This evidences that the global descriptors utilized in these approaches are capable of encoding 3D orientation information, but the described patterns are object specific, thus limiting the generalization ability of these methods to unseen objects. The performance of both PFS and GDR-Net also drops dramatically in the presence of unseen objects because the features extracted from 2D observations or 3D shapes remain strongly object dependent. Our method outperforms the competitors by a considerably large margin in the case of unseen objects, which demonstrates that the proposed patch-level image comparison framework generalizes better to unseen objects than previous works. This is because our use of patch-level similarities makes the network focus on local geometric attributes instead of high-level semantic information.

Table 1. Experimental results on LineMOD [13].

	Split #1		Split #2		Split #3		Mean		
	Seen	Unseen	Seen	Unseen	Seen	Unseen	Seen	Unseen	
Class. Acc. (%)	HOG [8]	39.57	41.26	30.24	32.65	36.19	35.19	35.33	36.37
	LD [36]	99.17	52.53	99.02	47.85	97.94	30.60	98.71	43.66
	NetVLAD [1]	100.00	68.36	100.00	52.30	100.00	47.22	100.00	55.96
	PFS [38]	-	-	-	-	-	-	-	-
	MPE [28]	98.75	57.25	83.29	69.98	97.73	87.57	93.26	71.60
	GDR-Net [34]	100.00	-	100.00	-	100.00	-	100.00	-
	Ours	100.00	97.90	99.44	92.47	98.03	88.93	99.16	93.10
Rota. Acc. (%)	HOG [8]	38.89	40.17	28.21	30.74	31.02	28.48	32.71	33.13
	LD [36]	94.50	8.63	89.57	12.47	91.47	5.22	91.85	8.77
	NetVLAD [1]	100.00	36.11	98.66	20.33	99.35	23.38	99.34	26.61
	PFS [38]	100.00	6.31	99.19	6.65	99.46	5.54	99.55	6.17
	MPE [28]	91.94	38.96	66.47	41.46	87.72	61.62	82.04	47.35
	GDR-Net [34]	99.89	4.61	99.28	4.82	99.31	5.02	99.49	4.82
	Ours	97.49	89.55	94.90	79.04	93.67	75.96	95.35	81.52

Table 2. Experimental results on LineMOD-O [3].

	Split #1		Split #2		Split #3		Mean		
	Seen	Unseen	Seen	Unseen	Seen	Unseen	Seen	Unseen	
Class. Acc. (%)	HOG [8]	0.60	0.60	0.23	0.23	36.20	36.20	12.34	12.34
	LD [36]	80.61	65.72	84.56	58.45	66.94	46.34	77.37	56.84
	NetVLAD [1]	85.27	56.46	74.37	47.73	90.33	66.32	83.32	56.84
	PFS [38]	-	-	-	-	-	-	-	-
	MPE [28]	83.29	56.07	55.26	45.08	70.72	57.55	69.76	52.90
	GDR-Net [34]	84.48	-	83.81	-	89.19	-	85.83	-
	Ours	83.22	83.99	78.69	74.42	82.44	75.06	81.45	77.82
Rota. Acc. (%)	HOG [8]	0.60	0.60	0.18	0.18	5.25	5.25	2.01	2.01
	LD [36]	32.21	6.25	26.56	3.26	24.57	4.57	27.78	4.69
	NetVLAD [1]	51.60	24.32	42.20	18.05	36.56	18.84	43.45	20.40
	PFS [38]	71.40	6.25	60.88	13.15	54.67	4.68	62.32	8.73
	MPE [28]	40.47	22.56	27.31	5.20	35.06	18.22	34.28	15.33
	GDR-Net [34]	63.37	3.12	55.31	2.97	49.91	2.39	56.20	2.83
	Ours	64.92	60.75	56.51	52.41	52.47	37.85	57.97	50.34

4.4 Experiments on T-LESS

To further evaluate the generalization ability to unseen objects, we conduct an experiment on T-LESS [14]. In this case, all deep learning approaches, including ours, were trained on LineMOD, and tested on the Primesense test scenes of T-LESS. As the objects’ appearance and shape in T-LESS are significantly different from the ones in LineMOD, this experiment provides a challenging benchmark to evaluate generalization. As in [15,29,28], we use $err_{vsd} \leq 0.3$ as a metric on this dataset. Note that we do not use the refinement module in [29,28] since it could be applied to all evaluated methods and thus is orthogonal to our contributions. As we concentrate on 3D object orientation estimation, we only consider the error of rotation matrices when computing err_{vsd} .

The results of all the methods are provided in Table 3. For this dataset, the 3D orientation estimates of all the previous deep learning methods are less accurate than those of the traditional approach, i.e., HOG. This shows that the models pretrained on LineMOD are unreliable when tested on T-LESS. By contrast, our method outperforms both hand-crafted and deep learning competitors. It evidences that our method can still effectively estimate 3D orientation for unseen objects even when the object’s appearance and shape entirely differ from those in the training data.

Table 3. Experimental results on T-LESS [14]. All deep learning methods were trained on LineMOD and tested on the Primesense test scenes of T-LESS. We use $err_{vsd} \leq 0.3$ as a metric.

Method	HOG [8]	LD [36]	NetVLAD [1]	PFS [38]	MPE [28]	GDR-Net [34]	Ours
Rota. Acc. (%)	74.22	24.19	56.46	17.92	66.88	11.89	78.73

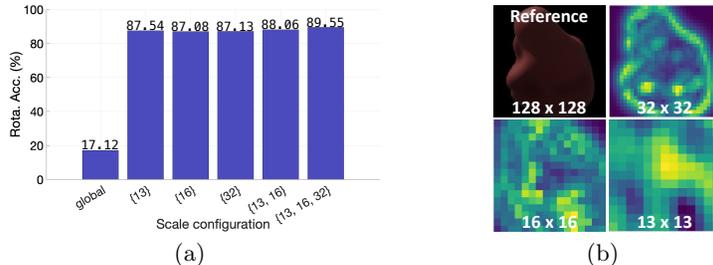


Fig. 5. Importance of local comparisons. (a) Results of our method with different scale configurations on the unseen objects of LineMOD Split #1. “Global” indicates a baseline that averages the smallest feature map into a global descriptor ($\mathbb{R}^{13 \times 13 \times C} \rightarrow \mathbb{R}^C$) for retrieval while maintaining the other components, except for the adaptive fusion, unchanged. (b) Feature maps used for retrieval.

4.5 Ablation Studies

Local Comparisons. One of the key differences between our method and existing works [36, 1, 28] is our use of the local comparisons during retrieval. Fig. 5(a) demonstrates the importance of local features in our framework. We start from a “global” baseline in which we average the smallest feature map along the spatial dimensions to form a global descriptor ($\mathbb{R}^{13 \times 13 \times C} \rightarrow \mathbb{R}^C$), which is used to compute the cosine similarity for retrieval. We keep the other components unchanged, except for the adaptive fusion. This baseline shows inferior generalization to unseen objects, while the performance significantly increases when local features are utilized. This observation indicates the importance of local comparisons for the unseen-object generalization. Moreover, the combination of multi-scale features also positively impacts the results, because it yields the ability to mix local geometric information at different scales, as illustrated by Fig. 5(b).

Adaptive Fusion. To analyze the importance of the adaptive fusion module in our method, we introduce the following three baselines. “Avg” consists of replacing the adaptive fusion with a simple averaging process, estimating the image similarity score by averaging all per-element similarities over the pairs of feature maps. Furthermore, “Sigmoid” and “Softmax” involve using only the sigmoid or softmax function in Eq. 3, respectively. As shown in Fig. 6(a), our approach yields an 8.7% increase in Rota. Acc. over “Avg”, which demonstrates the superiority of our adaptive fusion strategy. The reason behind this performance improvement is that our module assigns different confidence weights to the local similarities, as illustrated in Fig. 6(b), which makes it possible to distinguish the



Fig. 6. (a) Our adaptive fusion significantly outperforms a simple averaging strategy, and yields a boost over the “Sigmoid” and “Softmax” alternatives. (b) Confidence map depicting the weights in Eq. 3 obtained from the feature maps of the source and reference on the left. Yellow dots indicate local similarities with high confidence.

Table 4. Comparison between the fast clustering-based retrieval and greedy search. We report the 3D orientation accuracy and the test time on the unseen objects of LineMOD Split #1.

Method	Fast Retrieval	Greedy Search
Rota. Acc. (%)	89.55	95.93
Time (s)	0.42	30.74

useful information from the useless one. Moreover, the performance decreases (from 89.55% to 88.57% and 89.03%) when separately employing sigmoid and softmax functions in Eq. 3, which indicates the effectiveness of combining local and global context.

Fast Retrieval. We further conduct an experiment comparing the fast clustering-based retrieval with a greedy search strategy. The greedy search compares the source image with every reference in the database during retrieval. To achieve a fair comparison, we divide all references into different groups with a group size of 1024 and perform parallel estimation over the data in each group for the greedy search. As shown in Table 4, although the greedy search achieves a better 3D orientation estimation accuracy, the time consumption (30.74s) is not affordable in practice. Note that LineMOD contains 13 objects and we generate 10,000 reference images for each object. Therefore, the image comparison is performed 130,000 times for each source image in the greedy search, which results in an enormous time consumption. Note that we could not execute the comparisons w.r.t. all references in parallel because the NVIDIA Tesla V100 GPU could not store all the $130,000 \times 3$ feature maps. By contrast, our fast retrieval algorithm reduces the number of comparisons in two aspects: First, \mathbf{I}_{src} is only compared with the initial anchors of each object for category recognition, reducing the complexity from $O(NR)$ to $O(Nk_{ac})$; second, the clustering-based retrieval within the references that contain the recognized object for 3D orientation estimation only compares \mathbf{I}_{src} with dynamically updated anchors, which reduces the complexity from $O(R)$ to $O(k_{ac})$.

Number of References. Intuitively, we expect the number of reference images to be positively correlated with the 3D object orientation estimation accuracy

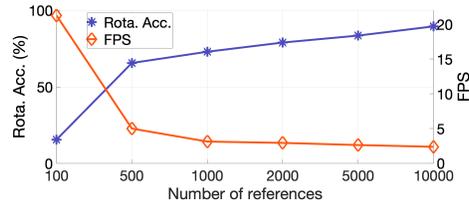


Fig. 7. Influence of the number of references. We vary the number of references for each object and report the results on the unseen objects of LineMOD Split #1.

Table 5. Influence of the individual components.

Local Norm.	Weighted infoNCE	Rota. Acc. (%)
✗	✗	85.91
✓	✗	89.25
✗	✓	87.14
✓	✓	89.55

while negatively correlated with the retrieval speed. To shed more light on the influence of the number of reference images, we evaluate our method with a varying number of references. As shown in Fig. 7, as expected, as the number of reference images increases, Rota. Acc. increases while FPS decreases. Therefore, one can flexibly adjust the number of references in practice according to the desired accuracy and efficiency.

Effectiveness of the individual components. Finally, we conduct an ablation study to further understand the importance of the other individual components in our method, i.e., local normalization and the weighted-infoNCE loss. As shown in Table 5, each of these two components has a positive impact on the 3D orientation estimation accuracy, and the optimal performance is achieved by leveraging both of them.

5 Conclusion

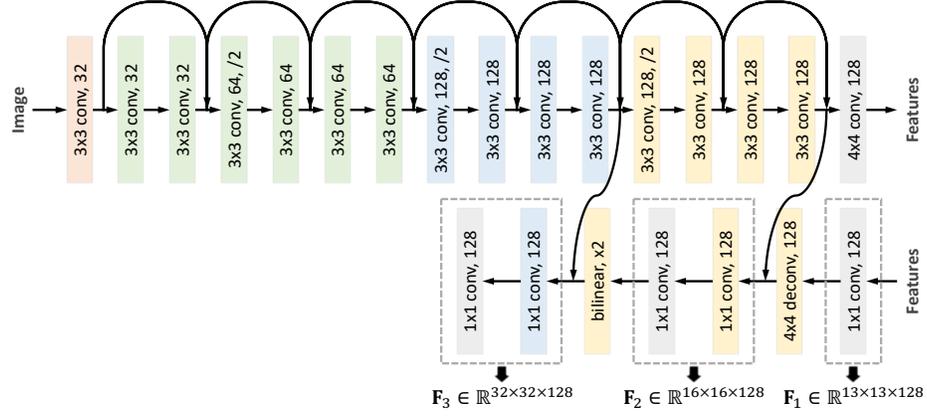
In this paper, we have presented a retrieval-based 3D orientation estimation method for *previously-unseen* objects. Instead of representing an image as a global descriptor, we convert it to multiple feature maps at different resolutions, whose elements represent local patches of different sizes in the original image. We perform retrieval based on patch-level similarities, which are adaptively fused into a single similarity score for a pair of images. We have also designed a fast clustering-based retrieval algorithm to speed up our method. Our experiments have demonstrated that our method outperforms both traditional and previous learning-based methods by a large margin in terms of 3D orientation estimation accuracy for unseen objects. In future work, we plan to extend our method to full 6D pose estimation of unseen objects.

References

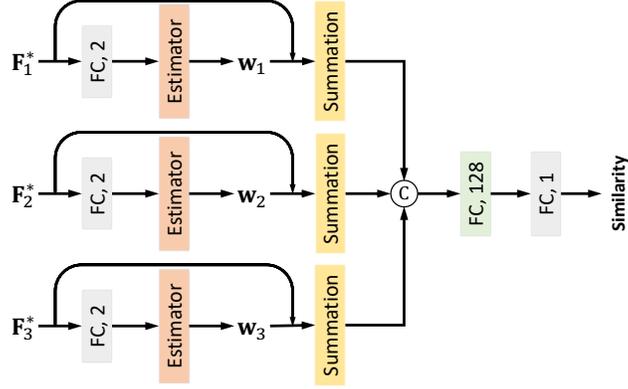
1. Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: Netvlad: Cnn architecture for weakly supervised place recognition. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 5297–5307 (2016)
2. Balntas, V., Doumanoglou, A., Sahin, C., Sock, J., Kouskouridas, R., Kim, T.K.: Pose guided rgb-d feature learning for 3d object pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3856–3864 (2017)
3. Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., Rother, C.: Learning 6d object pose estimation using 3d object coordinates. In: Proceedings of the European Conference on Computer Vision. pp. 536–551. Springer (2014)
4. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning. pp. 1597–1607. PMLR (2020)
5. Chen, W., Jia, X., Chang, H.J., Duan, J., Shen, L., Leonardis, A.: Fs-net: Fast shape-based network for category-level 6d object pose estimation with decoupled rotation mechanism. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1581–1590 (2021)
6. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3d object detection network for autonomous driving. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 1907–1915 (2017)
7. Collet, A., Martinez, M., Srinivasa, S.S.: The moped framework: Object recognition and pose estimation for manipulation. *The International Journal of Robotics Research* **30**(10), 1284–1306 (2011)
8. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. vol. 1, pp. 886–893. Ieee (2005)
9. Du, G., Wang, K., Lian, S., Zhao, K.: Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review. *Artificial Intelligence Review* **54**(3), 1677–1734 (2021)
10. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* **24**(6), 381–395 (1981)
11. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 3354–3361. IEEE (2012)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
13. Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N.: Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In: Asian Conference on Computer Vision. pp. 548–562. Springer (2012)
14. Hodan, T., Haluza, P., Obdržálek, Š., Matas, J., Lourakis, M., Zabulis, X.: T-less: An rgb-d dataset for 6d pose estimation of texture-less objects. In: Proceedings of the IEEE Winter Conference on Applications of Computer Vision. pp. 880–888. IEEE (2017)
15. Hodaň, T., Matas, J., Obdržálek, Š.: On evaluation of 6d object pose estimation. In: Proceedings of the European Conference on Computer Vision. pp. 606–619. Springer (2016)

16. Hu, Y., Fua, P., Wang, W., Salzmann, M.: Single-stage 6d object pose estimation. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 2930–2939 (2020)
17. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
18. Li, X., Wang, H., Yi, L., Guibas, L.J., Abbott, A.L., Song, S.: Category-level articulated object pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3706–3715 (2020)
19. Li, Y., Wang, G., Ji, X., Xiang, Y., Fox, D.: Deepim: Deep iterative matching for 6d pose estimation. In: Proceedings of the European Conference on Computer Vision. pp. 683–698 (2018)
20. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 3431–3440 (2015)
21. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. *Journal of Machine Learning Research* **9**(11) (2008)
22. Manuelli, L., Gao, W., Florence, P., Tedrake, R.: kpm: Keypoint affordances for category-level robotic manipulation. arXiv preprint arXiv:1903.06684 (2019)
23. Marchand, E., Uchiyama, H., Spindler, F.: Pose estimation for augmented reality: a hands-on survey. *IEEE Transactions on Visualization and Computer Graphics* **22**(12), 2633–2651 (2015)
24. Park, K., Mousavian, A., Xiang, Y., Fox, D.: 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. pp. 10710–10719 (2020)
25. Peng, S., Liu, Y., Huang, Q., Zhou, X., Bao, H.: Pvnet: Pixel-wise voting network for 6dof pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4561–4570 (2019)
26. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems* **30** (2017)
27. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 234–241. Springer (2015)
28. Sundermeyer, M., Durner, M., Puang, E.Y., Marton, Z.C., Vaskevicius, N., Arras, K.O., Triebel, R.: Multi-path learning for object pose estimation across domains. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 13916–13925 (2020)
29. Sundermeyer, M., Marton, Z.C., Durner, M., Brucker, M., Triebel, R.: Implicit 3d orientation learning for 6d object detection from rgb images. In: Proceedings of the European Conference on Computer Vision. pp. 699–715 (2018)
30. Tremblay, J., To, T., Sundaralingam, B., Xiang, Y., Fox, D., Birchfield, S.: Deep object pose estimation for semantic robotic grasping of household objects. arXiv preprint arXiv:1809.10790 (2018)
31. Vaze, S., Han, K., Vedaldi, A., Zisserman, A.: Generalized category discovery. arXiv preprint arXiv:2201.02609 (2022)
32. Wang, C., Martín-Martín, R., Xu, D., Lv, J., Lu, C., Fei-Fei, L., Savarese, S., Zhu, Y.: 6-pack: Category-level 6d pose tracker with anchor-based keypoints. In: Proceedings of the IEEE International Conference on Robotics and Automation. pp. 10059–10066. IEEE (2020)

33. Wang, C., Xu, D., Zhu, Y., Martín-Martín, R., Lu, C., Fei-Fei, L., Savarese, S.: Densfusion: 6d object pose estimation by iterative dense fusion. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3343–3352 (2019)
34. Wang, G., Manhardt, F., Tombari, F., Ji, X.: Gdr-net: Geometry-guided direct regression network for monocular 6d object pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 16611–16621 (2021)
35. Wang, H., Sridhar, S., Huang, J., Valentin, J., Song, S., Guibas, L.J.: 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. pp. 2642–2651 (2019)
36. Wohlhart, P., Lepetit, V.: Learning descriptors for object recognition and 3d pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3109–3118 (2015)
37. Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. arXiv preprint arXiv:1711.00199 (2017)
38. Xiao, Y., Qiu, X., Langlois, P.A., Aubry, M., Marlet, R.: Pose from shape: Deep pose estimation for arbitrary 3d objects. arXiv preprint arXiv:1906.05105 (2019)
39. Xu, D., Angelov, D., Jain, A.: Pointfusion: Deep sensor fusion for 3d bounding box estimation. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 244–253 (2018)
40. Zhou, Y., Barnes, C., Lu, J., Yang, J., Li, H.: On the continuity of rotation representations in neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5745–5753 (2019)
41. Zhu, M., Derpanis, K.G., Yang, Y., Brahmbhatt, S., Zhang, M., Phillips, C., Lecce, M., Daniilidis, K.: Single image 3d object detection and pose estimation for grasping. In: Proceedings of the IEEE International Conference on Robotics and Automation. pp. 3936–3943. IEEE (2014)



(a) Multi-Scale Feature Extraction



(b) Adaptive Fusion

Fig. 8. Network Architecture. \mathbf{F}_1 , \mathbf{F}_2 , and \mathbf{F}_3 indicate the three feature maps used to estimate local similarities. The estimator is used for the confidence map estimation.

A Network Architecture

Fig. A illustrates the network architectures of our multi-scale feature extraction module and adaptive fusion module. The multi-scale feature extraction module generates three feature maps of different sizes by default, i.e., $\mathbf{F}_1 \in \mathbb{R}^{13 \times 13 \times 128}$, $\mathbf{F}_2 \in \mathbb{R}^{16 \times 16 \times 128}$, and $\mathbf{F}_3 \in \mathbb{R}^{32 \times 32 \times 128}$. Given a pair of source image and reference image, the corresponding feature maps are paired to compute \mathbf{F}_1^* , \mathbf{F}_2^* , and \mathbf{F}_3^* , respectively. The local similarities are then adaptively fused into an image similarity score via our adaptive fusion module. Every convolution layer (conv) and fully connected layer (FC) is followed by ReLU, except for the ones in grey.

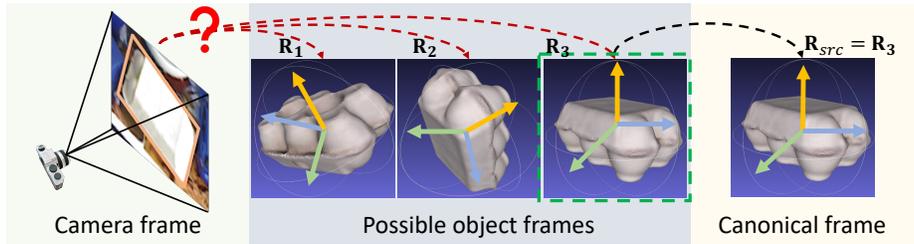


Fig. 9. Importance of a canonical frame. Different object frames correspond to different relative 3D rotations, which makes the 3D object orientation in the camera frame ill-defined. A canonical frame is required to uniquely define \mathbf{R}_{src} .

Table 6. Data splitting of LineMOD and LineMOD-O.

Dataset	Split #1	Split #2	Split #3
LineMOD	ape, benchvise camera, can	cat, driller duck, eggbox	glue, holepuncher iron, lamp, phone
LineMOD-O	ape, can	cat, driller, duck	eggbox, glue, holepuncher

B Canonical Frame

Fig. B shows the importance of a canonical frame to the 3D object orientation estimation. Specifically, the expected \mathbf{R}_{src} is the relative rotation between the camera frame and the object frame. Since the object can be placed in the real world with arbitrary poses, multiple possible object frames could exist. These object frames correspond to different relative rotations. Therefore, \mathbf{R}_{src} is ill-defined without a canonical frame. To address this issue, one could define a common canonical frame for all objects, but the shape variation among different objects makes this definition unreasonable. Consequently, we assume the 3D object models to be known in our experiments, which are employed to define the canonical frames dependently. Holding this assumption, we also use the 3D models to generate synthetic reference images for our retrieval-based 3D object orientation estimation.

C Experimental Setup

Table 6 shows the data splitting of LineMOD and LineMOD-O. The cropped data are assigned to three different groups according to the depicted objects.

Recall that the reference images are generated by rendering the corresponding 3D object model with different 3D orientation. The 3D orientation is formalized as a 3D rotation matrix. In our experiments, the 3D rotation matrix is randomly sampled, using the 6D representation [40]. Fig. C illustrates the distribution of

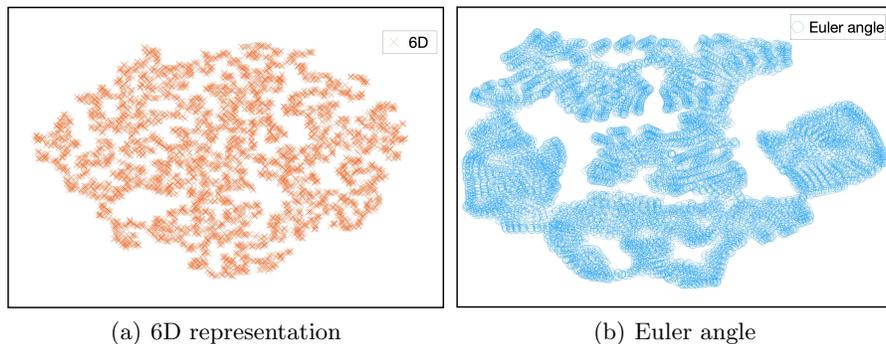


Fig. 10. Distribution of the sampled 3D rotation matrices. The 3D object orientation (3D rotation matrix) of the reference images is sampled using 6D representation [40] (a) and Euler angles [29] (b), respectively. The distribution is visualized by using t-SNE [21].

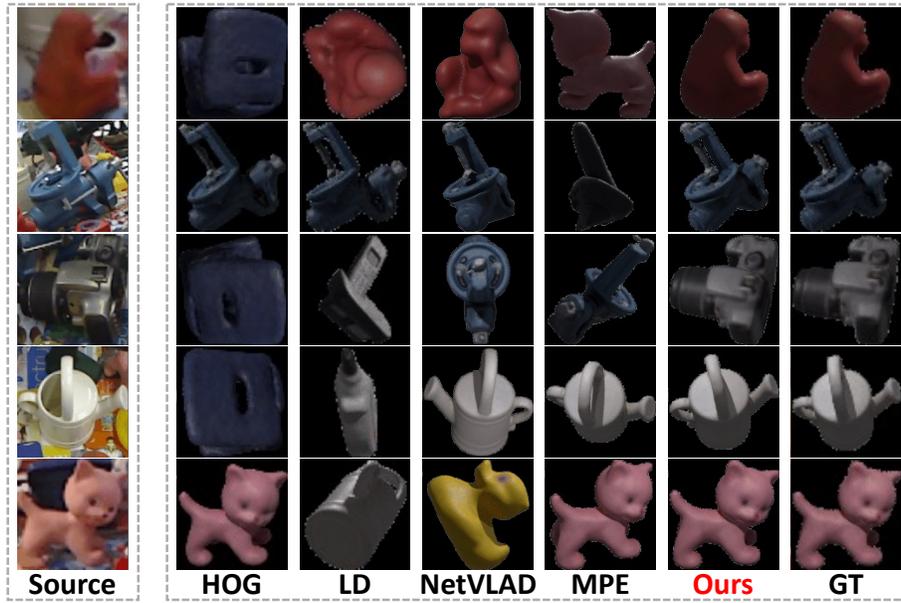
the sampled 3D rotation matrices, which is visualized by using t-SNE [21]. Compared with the samples using the representation of Euler angles [29] (Fig. 10(b)), the ones based on the 6D representation (Fig. 10(a)) are scattered in the sample space more evenly.

D Qualitative Results

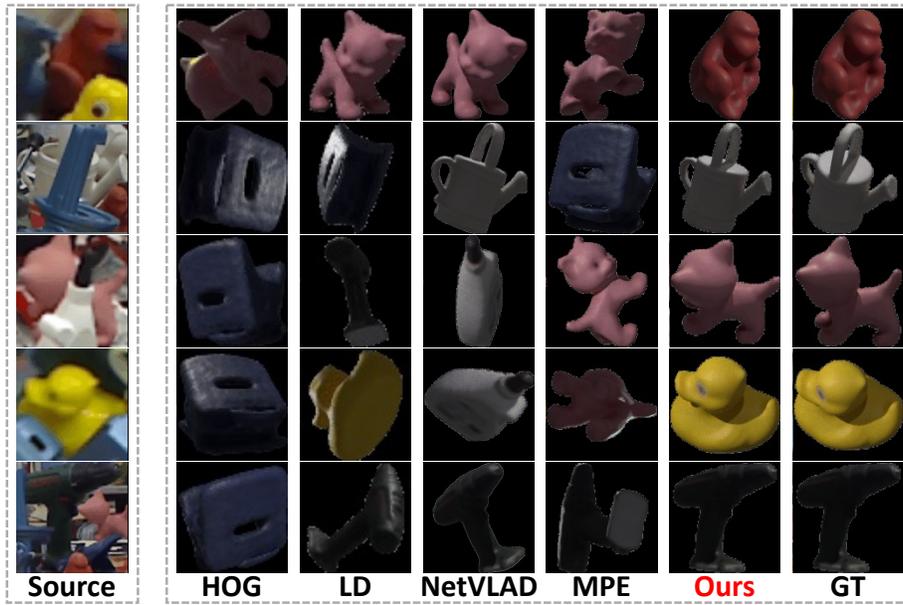
Fig. D shows some qualitative results in the presence of unseen objects on LineMOD and LineMOD-O. The images in the leftmost column are the real source images and the ones in the rightmost column are the most similar synthetic references. The other images are the retrieved results of the evaluated methods. One can observe that given some unseen objects, the previous approaches either select wrong objects or pick up the correct objects but with incorrect 3D orientation. By contrast, our method is capable of robustly retrieving the reference similar to the source image, and thus ensures the 3D orientation estimation accuracy for unseen objects.

E Quantitative Results

In the main paper, we assume that the object category is unknown on LineMOD and LineMOD-O, and the evaluated methods are used to both classify the object and estimate its 3D orientation. Therefore, the 3D object orientation estimation accuracy is related to the object classification accuracy (please refer to Eq. 9 in our main paper). To further evaluate the unseen-object generalization in the context of pure 3D object orientation estimation, we conduct another experiment on LineMOD and LineMOD-O, assuming the object category is known. In this



(a) LineMOD



(b) LineMOD-O

Fig. 11. Qualitative Results in the presence of unseen objects.

Table 7. Rota. Acc. (%) on LineMOD [13] in the case of unseen objects.

	Split #1	Split #2	Split #3	Mean
HOG [8]	71.43	66.24	53.75	63.81
LD [36]	14.39	19.46	13.32	15.72
NetVLAD [1]	42.84	38.04	41.31	40.73
MPE [28]	53.15	44.77	67.76	55.23
Ours	90.37	82.00	79.17	83.85

Table 8. Rota. Acc. (%) on LineMOD-O [3] in the case of unseen objects.

	Split #1	Split #2	Split #3	Mean
HOG [8]	34.51	34.92	28.01	32.48
LD [36]	7.85	4.30	6.98	6.38
NetVLAD [1]	37.33	24.31	23.37	28.34
MPE [28]	27.81	7.52	21.40	18.91
Ours	64.43	54.69	39.64	52.92

**Fig. 12. Failure cases on LineMOD [13] in the presence of unseen objects.**

case, Rota. Acc. is computed as

$$\text{Rota. Acc.} = \begin{cases} 1 & \text{if } d(\hat{\mathbf{R}}_{ref}, \mathbf{R}_{src}) < \lambda \\ 0 & \text{otherwise} \end{cases}. \quad (11)$$

We report Rota. Acc. (%) for unseen objects in Table 7 and Table 8. As this benchmark is less challenging, all methods yield better results compared with the ones reported in Sec. 4.3 of the main paper. In this context, our method still surpasses the competitors by a significantly large margin. This observation indicates the better distinctiveness of our method towards the 3D orientation of unseen objects. The superior results on LineMOD-O also evidence the robustness of our method to occlusions.

F Failure Cases

Fig. F illustrates some failure cases of our method on LineMOD. One can observe that there is a flipping issue in these cases. Taking the benchvise as an example, it is difficult to distinguish between our result and the ground-truth one, which makes this problem challenging. To address this issue, the network should

be able to extract fine-grained information. Ideally, our patch-level solution is more capable of capturing such information than image-level one. Therefore, in our future work, we plan to utilize hard example mining over the patch-level comparison to make our method pay more attention to fine-grained differences.