Towards Human-Level 3D Relative Pose Estimation: Generalizable, Training-Free, with Single Reference

Yuan Gao*, Yajing Luo*, Junhong Wang, Kui Jia, Gui-Song Xia

***Abstract*—Humans can easily deduce the relative pose of a previously unseen object, without labeling or training, given only a single query-reference image pair. This is arguably achieved by incorporating i) 3D/2.5D shape perception from a single image, ii) render-and-compare simulation, and iii) rich semantic cue awareness to furnish (coarse) reference-query correspondence. Motivated by this, we propose a novel 3D generalizable relative pose estimation method by elaborating 3D/2.5D shape perception with a 2.5D shape from an RGB-D reference, fulfilling the render-and-compare paradigm with an off-the-shelf differentiable renderer, and leveraging the semantic cues from a pretrained model like DINOv2. Specifically, our differentiable renderer takes the 2.5D rotatable mesh textured by the RGB and the semantic maps (obtained by DINOv2 from the RGB input), then renders new RGB and semantic maps (with back-surface culling) under a novel rotated view. The refinement loss comes from comparing the rendered RGB and semantic maps with the query ones, back-propagating the gradients through the differentiable renderer to refine the 3D relative pose. As a result, *our method can be readily applied to unseen objects, given only a single RGB-D reference, without labeling or training*. Extensive experiments on LineMOD, LM-O, and YCB-V show that our training-free method significantly outperforms the state-of-the-art supervised methods, especially under the rigorous `Acc@5/10/15°` metrics and the challenging cross-dataset settings. The codes are available at https://github.com/ethanygao/training-free_generalizable_relative_pose.**

***Index Terms*—3D Relative Pose Estimation, Differentiable Renderer, Zero-Shot Unseen Generalization, Single Reference, Label/Training-Free Refinement.**

I. Introduction

RECENT years have witnessed great progress in 3D object pose estimation [12, 13, 14, 22, 43, 44, 64, 67, 77, 78], which estimates the 3D rotation of an object depicted in a query RGB image. As a key to facilitating interaction with real-world objects, 3D object pose estimation attracts increasing attention from various areas including computer vision, virtual/augmented reality, robotics, and human-computer interaction [1, 50, 59]. To date, the community shows great interest in generalizable 3D object pose estimation [17, 44, 57, 76, 77, 78] owing to its wide applicability, which focuses on the generalization to previously unseen objects, preferably in a zero-shot manner[1].

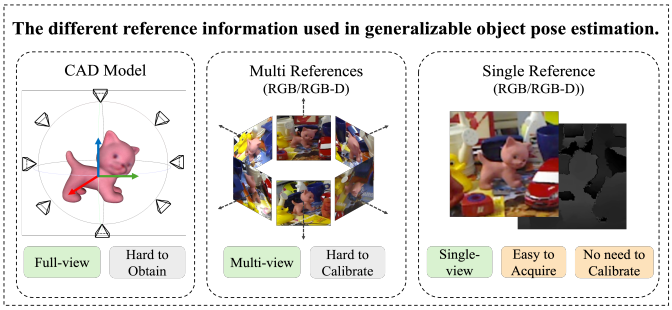

Fig. 1. **Generalizable object pose estimation with different references, i.e., *a CAD model, multiple images*, or *a single image*.** CAD models and multi-view references offer rich geometry details, however, scanning the precise CAD model and/or calibrating dense views for multiple images are laborious or even impossible for unseen objects in practice, such as augmented reality. We thus focus on estimating the relative pose w.r.t. a single-view reference following [77, 78], i.e., the relative pose between a reference-query pair, where we treat the reference pose as canonical without any calibration.

Existing generalizable 3D object pose estimation methods can be categorized according to how they exploit the reference information, i.e., using *a CAD model, multiple images*, or *a single image* as references, as shown in Fig. 1. Specifically, most existing methods leverage a 3D CAD model [7, 26, 44, 45, 55] or multiple images [17, 19, 34, 40, 48, 57, 76] for template matching or feature extraction, while the requirement of laborious 3D scanning (for the CAD-based methods) or multiple-image pose labeling (for most multi-image methods) severely limits their applicability.

On the other hand, recent methods propose to reframe the generalizable object pose estimation task as relative pose estimation between a query and a reference image from an unseen object, which is termed as *generalizable **relative** object pose estimation* [77, 78]. By treating the reference pose as canonical, estimating the relative pose between the reference-query pair successfully bypasses the laborious 3D scanning (of the CAD reference) or dense views calibration (of the multiple-image reference). However, existing methods rely on a large amount of well-labeled poses between the query-reference pairs to effectively train a neural network, thereby imposing the challenge of acquiring high-quantity training data [34, 76, 77, 78]. Moreover, the generalizability of some network-based methods may be impeded by the training data. Our empirical findings suggest that after pretrained on an

Y. Gao and G.-S. Xia are with the School of Artificial Intelligence, Wuhan University, Wuhan, China. E-mails: ethan.y.gao@gmail.com, guisong.xia@whu.edu.cn

Y. Luo is with the School of Computer Science, Wuhan University, Wuhan, China. E-mail: yajingluo@whu.edu.cn

J. Wang is with MoreFun Studio, Tencent Games, Tencent, Shenzhen, China. E-mail: junhongwang@tencent.com

K. Jia is with the School of Data Science, The Chinese University of Hong Kong, Shenzhen, China. E-mail: kuijia@cuhk.edu.cn

This work was supported by the National Natural Science Foundation of China (62306214, 62325111), and was also partially funded by the 2024 Shenzhen Science and Technology Major Project (202402002).

Corresponding authors: Yuan Gao, Gui-Song Xia.

* indicates equal contributions.

[1]We discuss the relatively easier instance- or category-level object pose estimation in the Related Work Sect. II-A and II-B, respectively.

TABLE I
THE TAXONOMY OF OUR METHOD IN **GENERALIZABLE POSE ESTIMATION**. FOR EACH COLUMN, WE ILLUSTRATE THE APPLICABILITY IN DESCENDING ORDER USING THE TEXT OF **BOLD**, UNDERLINED, AND NORMAL. WE ALSO INCLUDE THE HUMAN INTELLIGENCE AS A REFERENCE. THE STATE-OF-THE-ART METHODS USED IN OUR EXPERIMENTS ARE HIGHLIGHTED WITH THEIR NAME.

Method	Training	Label	Reference		Query	
			Modality	#Instance	Modality	#Instance
Human Intelligence	**training-free**	**label-free**	**RGB**	**single**	**RGB**	**single**
[7, 35, 45]	supervised	pose	CAD	multiple	RGB-D	**single**
[26, 42, 44, 47, 55]	supervised	pose	CAD	multiple	**RGB**	**single**
[19, 48]	supervised	pose	RGB-D	multiple	RGB-D	**single**
RelPose++ [34], [17, 40, 57, 76]	supervised	pose	**RGB**	multiple	**RGB**	**single**
LoFTR [58]	supervised	pose+depth	**RGB**	**single**	**RGB**	**single**
3DAHV [78], DVMNet [77]	supervised	pose	**RGB**	**single**	**RGB**	**single**
ZSP [16]	**training-free**	**label-free**	RGB-D	**single**	RGB-D	multiple
Ours	**training-free**	**label-free**	RGB-D	**single**	**RGB**	**single**

external large-scale dataset such as Objaverse [10], the current state-of-the-art methods [77, 78] require in-dataset finetuning[2] before testing on unseen objects within the dataset, which might potentially hinder their cross-dataset generalizability.

In this context, we work towards universally applicable zero-shot 3D generalizable relative pose estimation, where i) the object is agnostic/unseen from a cross-dataset, ii) only a single RGB-D image is available for reference without a 3D CAD model or multi-view images, and iii) the ground-truth (relative) pose label is not available. In other words, we aim to establish a novel 3D generalizable (in terms of both objects and datasets) relative pose estimation method given only one reference and one query image, without labeling or training. This is extremely challenging due to the mixture of *incomplete shape information* and *missing reference-query correspondence*, which leads to a severely degraded optimization problem.

Our method is inspired by the fact that humans can easily infer the relative pose under the aforementioned rigorous setting, even with large pose differences or severe occlusions. We hypothesize that such intelligence is accomplished through i) perceiving 3D/2.5D shapes from a single image, ii) conducting render-and-compare simulations via imagination, and iii) understanding rich semantic cues of the object. For example, given two viewpoints of an unseen animal, humans are able to infer the 3D/2.5D shape of that animal, then identify the correspondences of the animal eyes, noses, ears, etc, and finally rotate and render the 3D/2.5D model until its projection matches the other view. Note that the semantic cues have the potential to deal with the (self-) occluded missing parts, thus enhancing the comparison process, e.g., an animal tail can be simply ignored in the render-and-compare simulations if it only appears in one image and is (self-) occluded in the other.

The above analysis motivates us to break down our difficulties and fulfill those three requirements. Concretely, we achieve this by formulating a label/training-free framework through an off-the-shelf differentiable renderer following the render-and-compare paradigm. Our input shape to the differentiable renderer is an RGB- and semantic-textured 2.5D mesh of the reference (avoiding the difficult 3D hallucination of an unseen object). Based on this, we construct a pose refinement framework, where the differentiable renderer takes an initial pose to render projections, then back-propagates the gradients from the projection loss (between the rendered and the query images) to refine the initial pose.

Specifically, our method starts with an RGB-D reference and an RGB query, where their semantic maps can be obtained by leveraging an advanced pretrained model DINOv2 [46] with the RGB inputs[3]. We leverage an easy-to-use differentiable renderer nvdiffrast [27], which takes the RGB- and semantic-textured 2.5D mesh of the reference as input, then renders new RGB and semantic maps (with back-surface culling) under a novel rotated view. The pose refinement loss comes from comparing the rendered RGB and semantic maps with the query ones, which flows the gradients through the differentiable renderer to refine the 3D relative pose. As a result, our method can be readily applied to unseen objects from an arbitrary dataset without labeling or training, and naturally generalizes to cross-dataset scenarios.

In summary, we propose a novel 3D generalizable relative pose estimation method, which takes only an RGB-D reference and an RGB query pair, without requiring the ground-truth pose labels or training. We achieve this by formulating a pose refinement framework via an off-the-shelf differentiable renderer under the render-and-compare paradigm. Our method does not involve training a network, which naturally possesses zero-shot generalizability in terms of both unseen objects and datasets. We conducted extensive experiments on LineMOD [20], LM-O [2] and YCB-V [70] datasets. The results from our training-free method exhibit significant improvement over the state-of-the-art supervised methods, e.g., for `Acc@15°` metric on the LineMOD dataset [20] and the YCB-V dataset [70], our label- and training-free method outperforms the supervised state-of-the-art results by **29.98%** and **14.28%**, respectively. Our contributions are three-fold:

- We propose a novel and simple relative pose estimation method that naturally generalizes to unseen objects,

[2]The in-dataset finetuning denotes that the finetune set comes from the same dataset with the testing set, while not including the testing objects.

[3]Note that our method possesses the potential of using only an RGB reference, please see the discussion in Sect. I-A (Applicability) and Sect. VI (Limitations and Future Works) for more details. Moreover, our method works reasonably well even without the DINOv2 semantic maps on the LineMOD dataset, as illustrated in Table V.

without the need for ground truth pose labels or neural network training.
- Our method is optimized by an off-the-shelf differentiable renderer and thus training-free. This eliminates the training-data dependency in supervised CNN/ViT-based methods, thus ensuring inherently robust generalizability.
- Our method employs a render-and-compare framework leveraging 2.5D meshes, thereby avoiding the challenging 3D hallucination of unseen objects. Building upon this, semantic features (such as those from DINOv2) can be integrated as texture maps (via PCA dimensionality reduction) into the render-and-compare process.

A. Taxonomy and Applicability of Our Method

Taxonomy. The taxonomy of our methods in generalizable pose estimation, in terms of `training`, `labeling`, as well as the `modality` and the `number of required instances` of the *reference* and the *query* images, is illustrated in Table I. Our method falls under the category of *label/training-free* with a *single RGB query* and a *single RGB-D reference*.

Applicability. Among Table I, the proposed method shares the closest setting to the human intelligence on relative pose estimation that is able to generalize to unseen objects from an arbitrary dataset, with only an additional one-time-collection depth map for the reference image.

Our method arguably possesses better applicability compared to the state-of-the-art methods summarized in Table I, as i) unlike supervised in-dataset state-of-the-art methods, our method does not require ground truth (GT) pose annotations for training, where obtaining a large number of GT poses is arguably more challenging than acquiring a single reference depth map. ii) Our method requires only a one-time reference depth annotation, which can be collected and fixed in advance. Moreover, depth sensors are commonly used in our primary application domain, i.e., robotics. We have testified in supplementary material Table S1 that our method can still deliver good estimations with an imprecise depth map, simulating noisy depth sensors. iii) Our training-free method naturally generalizes to unseen objects because it does not involve training a neural network, and thus is training-data independent. In contrast, supervised state-of-the-art methods typically perform less satisfactorily in cross-dataset scenarios, as they suffer from training-data dependency, leading to generalization issues given different training and evaluation datasets.

To further examine the potential of fully distilling human intelligence, we conducted ablation experiments in the supplementary material by substituting the GT depth with predictions from the state-of-the-art Depth Anything v2 [73]. We note that to preserve the object shape, our method requires the ***metric*** depth, meaning the estimated depth z and spatial dimensions x, y should share the same unit of measurement (e.g., both in meters). However, as shown in Table S2-S4 in supplementary material, the off-the-shelf metric depth estimator occasionally fails to generalize across different datasets, likely due to an imprecisely recovered depth scale caused by varying camera parameters and/or diverse objects across different training and evaluation datasets. We note that a generalizable metric depth estimator would alleviate this issue, but training a generalizable metric depth estimator is beyond the scope of, and may introduce distractions to, our current focus.

Finally, our method also incorporates the segmentation maps of both query and reference objects as input, which can be obtained by pretrained segmentation models such as SAM [25], FastSAM [80] and Grounded SAM [52]. We chose not to delve into these segmentation techniques extensively either, for the same sake of minimizing potential distractions.

II. Related Work

A. Instance-level 6D Pose Estimation

Current object pose estimation can be categorized into instance-level, category-level, and generalizable methods based on different problem formulations. For instance-level methods, there are roughly three categories: direct regression-based, correspondence-based, and refinement-based. Direct regression-based methods [3, 23, 51, 60, 70] predict the object pose directly through a neural network. Correspondence-based methods [11, 18, 21, 32, 33, 49, 56, 65, 72, 74] estimate the 2D-3D/3D-3D correspondence between the 2D images and 3D object models, followed by PnP solvers [30] to calculate 6D poses. Additionally, refinement-based methods [31, 39, 71] incorporate refinement-based steps to improve the prediction performance. However, instance-level methods are trained on instance-specific data and rely heavily on CAD models to render numerous training data. Consequently, their application is limited to the objects on which they were trained.

B. Category-level 6D Pose Estimation

In category-level methods, the test instances are not seen during training but belong to known categories. Most methods achieve this by either alignment or directly regressing. Alignment-based methods [8, 29, 36, 61, 63, 66] first propose a Normalized Object Coordinate Space (NOCS) [66] as a canonical representation for all possible object instances within a category. A network is then trained to predict the NOCS maps and align the object point cloud with the NOCS maps using the Umeyama algorithm [62] to determine the object pose. This method typically constructs the mean shape of specific categories as shape priors using offline categorical object models, and the networks are trained to learn deformation fields from the shape priors to enhance the prediction of NOCS maps. In contrast, directly regressing methods [6, 9, 37, 38, 41] avoid the non-differentiable Umeyama algorithm and often focus on geometry-aware feature extraction. For instance, CASS [6] contrasts and fuses shape-dependent/pose-dependent features to predict both the object's pose and size directly. Fs-net [9] leverages 3D Graph Convolution for latent feature extraction, and designs shape-based and residual-based networks for pose estimation. However, while category-level methods strive to address different instances within the same category, their capacity to predict the poses of objects from entirely new categories remains limited, highlighting the ongoing need to broaden the scope of object pose estimation to encompass unfamiliar objects.

C. Generalizable 6D Pose Estimation

Generalizable algorithms aim to enhance the generalizability of unseen objects without the need for retraining or finetuning. Methods in this category can be classified as CAD-based [4, 7, 26, 42, 44, 45, 47, 55] or multi-view reference-based [17, 19, 40, 48, 57, 68].

For CAD-based approaches, CAD models are often used as prior knowledge for direct feature matching or template generation. In particular, ZeroPose [7] performs point feature extraction for both CAD models and observed point clouds, utilizing a hierarchical geometric feature matching network to establish correspondences. Following ZeroPose, SAM-6D [35] proposed a two-stage partial-to-partial point matching model to construct dense 3D-3D correspondence effectively. Instead, Template-Pose [44] utilizes a CAD model to generate a collection of templates and selects the most similar one for a given query image. Similarly, OSOP [55] renders plenty of templates and estimates the 2D-2D correspondence between the best matching template and the query image to solve the object pose. MegaPose [26] proposed a coarse network to classify which rendered image best matches the query image and generate an initial pose. Subsequently, multi-view renderings of the initial pose are produced, and a refiner is trained to predict an updated pose.

Multi-view reference-based methods can be further divided into feature matching-based and template matching-based approaches. For the former, multi-view reference-based feature matching methods mainly aim to establish 2D-3D correspondences between the RGB query image and sparse point cloud reconstructed by reference views or 3D-3D correspondences between the RGB-D query and RGB-D reference images. For instance, FS6D [19] designed a dense prototype matching framework by extracting and matching dense RGBD prototypes with transformers. After the correspondence is established, Umeyama [62] algorithms are utilized for pose estimation. OnePose/OnePose++ [17, 57] apply the Structure from Motion (SfM) method to reconstruct a sparse point cloud of the unseen object using all reference views. They then employ an attention-based network to predict the correspondence between 2D pixels and the reconstructed point clouds to estimate the object pose. For the latter, multi-view references can be reviewed as templates for retrieval when plenty of views exist, or used to reconstruct the 3D object models for template rendering, similar to the CAD-based methods. Gen6D [40] selects the closest reference view for the query image, and then refines the pose through the 3D feature volume constructed from both the reference and query images. Notably, Gen6D requires more than 200 reference images for initial pose selection. On the contrary, LatentFusion [48] reconstructs a latent 3D representation of an object to present an end-to-end differentiable reconstruction and rendering pipeline, and then estimates the pose through gradients update. Since a 3D object representation can be reconstructed utilizing the multi-view information, FoundationPose [69] proposed a unified framework to support both CAD-based and multi-view supported setups. When no CAD model is available, they leverage multi-view references to build a neural implicit representation, which is then used for render-and-compare.

D. Generalizable Relative Pose Estimation

Recent methods [34, 76, 77, 78] highlight the importance of formulating object pose estimation as a relative pose estimation problem. Specifically, [77, 78] address situations where only a single-view reference image is available. [78] evidence that some state-of-the-art feature matching approaches [16, 54, 58] fail to generate reliable correspondence between the reference-query pair, while energy-based methods [34, 76] struggles to capture 3D information. Instead, 3DAHV [78] introduces a framework of hypothesis and verification for generating and evaluating multiple pose hypotheses. Based on that, DVMNet [77] directly lifts the 2D image features to 3D voxel information in a hypothesis-free way, computing the relative pose estimation in an end-to-end fashion by aligning the 3D voxels.

III. Method

Following the render-and-compare paradigm, current generalizable pose estimation methods often rely on rotatable 3D CAD models or well-calibrated multi-view images, imposing challenges to acquire the 3D CAD models or expensive pose calibration, especially for previously unseen objects. We instead focus on the generalizable **relative** pose estimation formulated in the pioneering works [77, 78], which aims to estimate the relative pose between a reference-query pair, using only a single reference with an arbitrary pose as canonical (without calibration). Our method differs from [77, 78] in not requiring labeled relative pose to train an estimation network.

A. Overview

Taking an RGB query and an RGB-D reference as input, our method establishes a refinement optimization under the render-and-compare framework, by leveraging a 2.5D (i.e., RGB-D) shape of the reference, a pair of semantic maps for both the query and the reference acquired by a pretrained DINOv2 model [46] along with the corresponding RGB maps, and a differentiable renderer to backpropagate the gradients. Note that the 2.5D shape is exploited due to the inherent difficulty of accurately hallucinating the 3D shape of unseen objects when relying solely on a single RGB-D image. This challenge further complicates the task of relative pose estimation, as the hallucinated 3D shape must align precisely with the query to achieve a successful estimation.

Formally, by using *subscript* to denote query or reference, our method starts with an RGB pair I_r and I_q for both reference and query, as well as a depth map D_r for the reference. We proposed to estimate the relative pose between I_r and I_q, assisted by D_r. To this end, we first infer the semantic maps S_r and S_q from I_r and I_q exploiting a pretrained DINOv2 model [46]. Then, we construct a 2.5D mesh model M_r for the reference object based on D_r, to formulate an RGB and semantic maps textured 2.5D mesh $\mathcal{M}_r = \{M_r, I_r, S_r\}$. Subsequently, the textured 2.5D reference mesh $\mathcal{M}_r$ is rotated with an (arbitrary) initial pose P by a differentiable renderer [27] to generate novel $I_r(P)$ and $S_r(P)$. Finally, the generated $I_r(P)$ and $S_r(P)$ are compared with the query I_q and S_q, producing a refinement loss and consequently back-propagate gradients to P through the differentiable renderer.

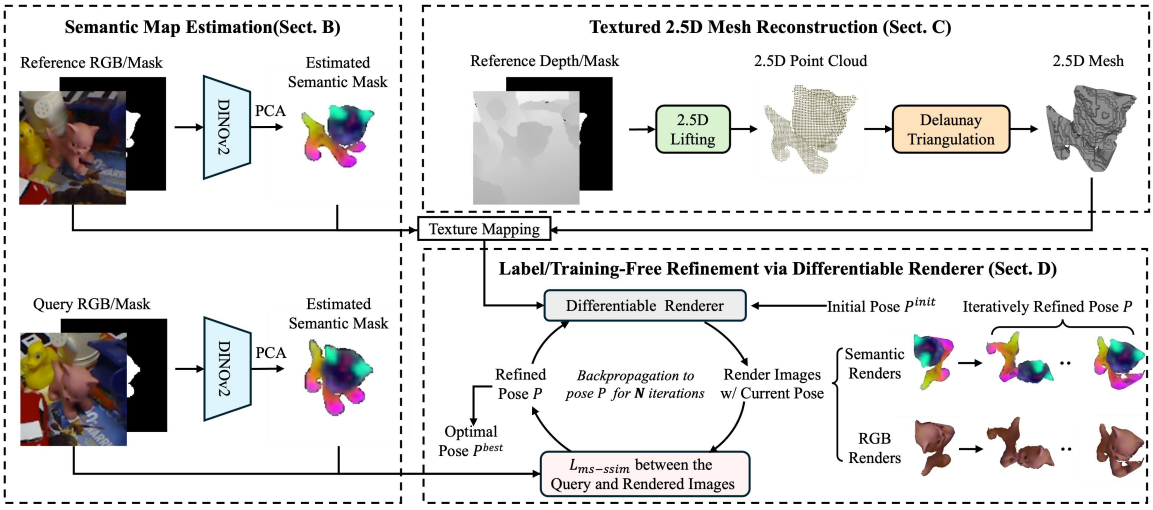

Fig. 2. **The overview of the proposed method.** Given an RGB-D reference and an RGB query, we extract the semantic maps from a pretrained DINOv2 model [46] for both reference and query. Then, the reference 2.5D front-surface mesh is reconstructed by the depth input without hallucination, which is subsequently texture-mapped by its RGB and semantic images. By leveraging a differentiable renderer [27], we generate the rendered RGB and semantic maps using the textured 2.5D reference mesh under a novel view/pose. Finally, the rendered RGB and semantic maps are compared to their query counterparts, producing losses and back-propagating the gradients through the differentiable renderer to refine the relative pose.

Our method operates the render-and-compare procedure in a self-supervised and network-free manner, without labeling or training.

The overview of the proposed method is illustrated in Fig. 2. We detail the comprising elements of our method in the following sections, i.e., **semantic map estimation** in Sect. III-B, **textured 2.5D mesh reconstruction** in Sect. III-C, and **label/training-free refinement via differentiable renderer** in Sect. III-D.

B. Semantic Map Estimation

In order to estimate the relative pose, human intelligence may unconsciously infer the semantics of the reference-query pair. Subsequently, coarse correspondence can be established with those semantics, resulting in three-fold benefits: i) it helps to filter out the large non-overlapped part under a substantial pose difference, ii) alleviates the influence of occlusions, and iii) eases the degraded optimization of the relative pose estimation.

Benefit from the rapid development of large pretrained models, an elegant off-the-shelf semantic feature extractor is available as DINO/DINOv2 [5, 46], which shows great zero-shot generalizability to diverse (even texture-less) objects (see Fig. 3 for some examples). We thus incorporate the off-the-shelf DINOv2 model [46] to acquire the rich semantics of the input unseen objects.

Specifically, we utilize DINOv2 [46] as the semantic feature extractor $\mathbf{\Phi}(\mathbf{x})$, which takes an RGB image I to produce a set of semantic features $F \in \mathbb{R}^{w \times h \times d}$. In order to texture F to the 2.5D model and facilitate the novel pose rendering, we use the principal component analysis (PCA) to reduce the dimension of F from d to 3, obtaining a semantic map S:

$$S = \texttt{PCA}(\mathbf{\Phi}(I)), \quad \texttt{PCA} : \mathbb{R}^{w \times h \times d} \rightarrow \mathbb{R}^{w \times h \times 3}. \tag{1}$$

By feeding Eq. (1) with I_q and I_r, we can obtain the semantic maps for the query and the reference, S_q and S_r, respectively. To ensure the semantic consistency between reference and query images, we first calculate and fix the PCA transformation using the reference image, then apply this transformation to all the query images.

C. Textured 2.5D Mesh Reconstruction

In this section, we reconstruct a rotatable 2.5D model of the reference given its depth map D_r, which is subsequently used to generate novel renderings through the differentiable renderer. Note that our design avoids the challenging 3D hallucination of an unseen object from the depth map, as the hallucinated 3D shape must consistently align with the query for relative pose estimation.

Specifically, given the depth map D_r of the reference, we lift the coordinates of the image plane into the 3D space and obtain the front surface 2.5D point clouds $X_r \in \mathbb{R}^{N \times 3}$. We then reconstruct the corresponding 2.5D mesh M_r from X_r, to facilitate the rasterization in the renderer. Since the xy coordinates of X_r are sampled regularly from the 2D grids, reconstructing M_r from X_r can be easily achieved by the Delaunay triangulations [28]. Finally, we texture M_r with both color and semantic maps, obtaining $\mathcal{M}_r = \texttt{TextMap}(M_r, I_r, S_r)$ for rendering under novel poses.

Note that as discussed in Sect. I-A (Applicability), our method possesses the potential of using only an RGB reference and estimating an imprecise depth map exploiting an off-the-shelf generalizable depth estimator. Good estimation is validated in Sect. S2 of the supplementary material given an imprecise and noisy depth. We leave training a generalizable depth estimator as our future work to avoid possible distractions in this paper.

D. Label/Training-Free Refinement via Differentiable Renderer

Our last module of label/training-free refinement is constructed by a differentiable renderer, which takes the textured 2.5D reference mesh $\mathcal{M}_r$ and a pose P as input, then renders

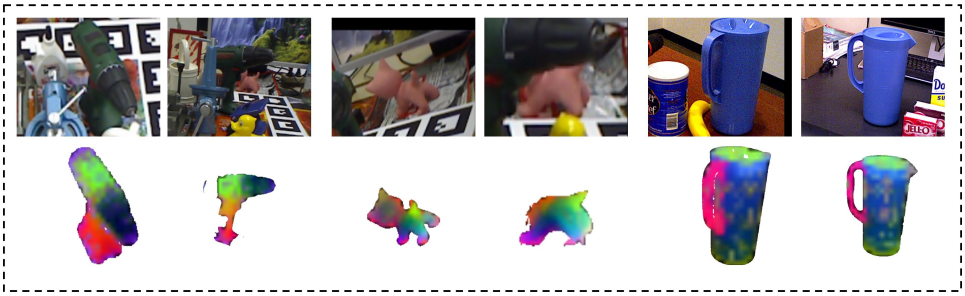

Fig. 3. **Illustration of the semantic maps estimated by DINOv2 [46]**, demonstrating promising zero-shot performance even for texture-less objects.

a novel RGB image and a novel semantic map under the view P. By implementing the pose P as a random variable, the render-and-compare/reprojection loss can be back-propagated directly to P, ensuring the label/training-free and zero-shot unseen generalization merits of our proposed method.

Formally, by assuming a perspective camera, we leverage a recent differentiable renderer `nvdiffrast` [27], denoted as $\mathcal{R}$, to generate novel RGB and semantic maps, $I_r(P)$ and $S_r(P)$, from the textured 2.5D reference mesh $\mathcal{M}_r$, an arbitrary pose P, and the camera intrinsic K[4]:

$$I_r(P), S_r(P) = \mathcal{R}(P, \mathcal{M}_{\mathrm{r}}, K) \tag{2}$$

Back Surface Culling. As the reconstructed mesh is only 2.5D representing the front surface, it is crucial to conduct the back-surface culling during the rendering to filter out the incorrect back-facing polygons. Specifically, for every triangle of the mesh, we first calculate the dot product of their surface normal and the camera-to-triangle (usually set to $[0, 0, 1]$) and then discard all triangles whose dot product is greater or equal to 0 [75]. Please also see the ablation with and without the back-surface culling in Table V.

Finally, the pose P can be optimized to align the rendered $I_r(P)$ and $S_r(P)$ with the query I_q and S_q, with the re-projection loss calculated by:

$$L(P) = L_1\left\{I_r(P); I_q\right\} + L_2\left\{S_r(P); S_q\right\}, \tag{3}$$

where $L(P)$ is the final loss to optimize the pose P, and we implement both losses by the multi-scale structural similarity (MS-SSIM) [79] as the following:

$$L_1 = 1 - \text{ms-ssim}\left\{I_r(P); I_q\right\}, \tag{4}$$

$$L_2 = 1 - \text{ms-ssim}\left\{S_r(P); S_q\right\}, \tag{5}$$

Equation (3) enables us to optimize P simply by gradient descent.

Initialization. As revealed in the majority of prior arts [26, 31, 44, 48], a good initialization significantly boosts the performance of the render-and-compare framework.

To this end, we implement our initialization by evenly sampling candidate poses on a sphere and chasing the best one. Specifically, we first sample m viewpoints (azimuth and elevation angles) uniformly using a Fibonacci lattice [15], then uniformly sample n in-plane rotation angles for each viewpoint, producing $t = m{*}n$ poses as the initializing candidates.

[4]The camera intrinsic K can be obtained from the image EXIF information.

Algorithm 1 Generalizable Label/Training-Free Refinement

Input: Reference RGB and depth I_r, D_r; query RGB I_q; differentiable renderer $\mathcal{R}$; pretrained DINOv2 model $\mathbf{\Phi}$, iteration quota N, learning rate α, camera intrinsic K.

$S_q \leftarrow \texttt{PCA}(\mathbf{\Phi}(I_q))$, $S_r \leftarrow \texttt{PCA}(\mathbf{\Phi}(I_r))$
$M_r \leftarrow \texttt{DelaunayTriangulations}(I_r, D_r)$
$\mathcal{M}_r \leftarrow \texttt{TextMap}(M_r, I_r, S_r)$
▷ Sampling Poses for Initialization
$\left\{P^1, P^2, ..., P^n\right\} \leftarrow \texttt{Uniformly\_sampling()}$
$\mathbf{P} = \left\{P^1, P^2, ..., P^n\right\}$
$I_r(\mathbf{P}), S_r(\mathbf{P}) \leftarrow \mathcal{R}(\mathbf{P}, \mathcal{M}_{\mathrm{r}}, K)$
$P^{\text{init}} = \arg\min_{P^i \in \mathbf{P}} L_1\left\{I_r(P^i); I_q\right\} + L_2\left\{S_r(P^i); S_q\right\}$
▷ Label/Training-Free Refinement via Diff. Renderer
$P \leftarrow P^{\text{init}}$
for $i < N$ **do**
  $I_r(P), S_r(P) \leftarrow \mathcal{R}(P, \mathcal{M}_{\mathrm{r}}, K)$
  $L(P) = L_1\left\{I_r(P); I_q\right\} + L_2\left\{S_r(P); S_q\right\}$
  $P \leftarrow \texttt{GradientDescent}(L(P), \alpha)$
end for

Output: P

By rendering both RGB and semantic maps of those candidate poses, we are able to calculate the re-projection loss by Eq. (3) (without back-propagation in this phase) and choose the pose with the minimal loss as our initialization P^{init}.

Given the initialized pose P^{init}, we perform N iterations with gradient back-propagation to carry out the label/training-free refinement via the differentiable renderer. Our algorithm is detailed in Algorithm 1.

IV. Experiments

In this section, we extensively validate our method on benchmark datasets including the LineMOD [20], YCB-V [70], and LineMOD-Occlusion (LM-O) [2] datasets. We detail the experimental setup in the following.

A. Experimental Setups

State-of-the-art Methods for Comparison. As shown in Table I, there does not exist a method applying the challenging setting of label/training-free and a single reference-query pair like ours. Therefore we choose the state-of-the-art methods that share the closest experimental setups, which are **ZSP** [16], **LoFTR** [58], **RelPose++** [34], **3DAHV** [78], and **DVMNet** [77]. Specifically, for **ZSP**, though it was originally proposed

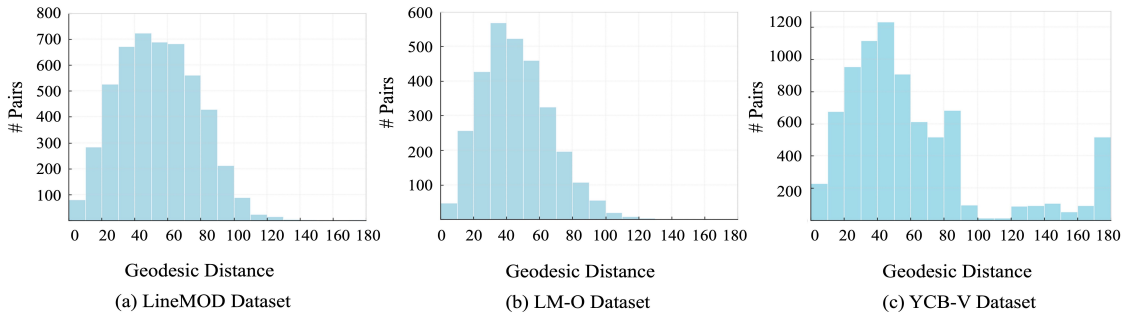

Fig. 4. **Histograms of the geodesic distance between the sampled reference-query pairs.** The in-plane rotation is included in calculating the histograms.

to process multiple queries, it is able to accept one RGB-D query as input. We report its performance based on the single RGB-D query and single RGB-D reference pair. For **LoFTR**, we use its pretrained weights released by the authors [58]. The weights of **DVMNet**, **3DAHV**, and **RelPose++** are retrained on-demand to achieve their best performance (for the details, see the following *Benchmark Experiments*, and the table captions of Table II, Table III and Table IV).

Datasets. The experiments are carried out on three benchmark object pose estimation datasets, i.e., LineMOD dataset [20] comprises 13 real objects, each depicting a single low-textured object on varying lighting conditions with approximately 1,200 images. LineMOD-Occlusion (LM-O) [2] consists of 1,214 images of the 8 occluded objects, extracted from the LineMOD dataset, the average visible fraction of objects in LM-O is 79.45%. YCB-V [70] encompasses over 110,000 real images featuring 21 objects characterized by severe occlusion and clutter, it exhibits an average visible object fraction of 87.15%.

Evaluation Metric. Following [77, 78], we report `mean angular error` across sampled reference-query pairs. We also evaluate on important metrics of `Acc@5/10/15/30°`, i.e., the percentage of the predictions that are within 5/10/15/30°, which can be more rigorous (e.g., `Acc@5°`) and better characterize the performance. The degree of the pose difference between the ground truth R_{gt} and the predictions $\hat{R}$ is calculated by the geodesic distance D:

$$D = \arccos\left(\left(\text{tr}(\Delta R_{gt}^T \Delta \hat{R}) - 1\right)/2\right)/\pi \tag{6}$$

Benchmark Experiments. The in-dataset networks of the state-of-the-art DVMNet, 3DAHV, and RelPose++ methods need to be trained on the leave-out subset which comes from the same dataset as the testing subset but does not include the testing objects. For a fair comparison, on the LineMOD dataset, we follow the experiments in DVMNet [77] and 3DAHV [78] to evaluate 5 objects (i.e., benchvise, camera, cat, driller, duck). For the YCB-V experiments, we design a similar training protocol to enable the comparison with DVMNet, 3DAHV, and RelPose++, where we randomly sample 8 objects (i.e., tuna_fish_can, pudding_box, banana, pitcher_base, mug, power_drill, large_clamp, foam_brick) for evaluation, leaving the remaining 13 objects to train these three methods. Following DVMNet [77], we evaluate 3 unseen objects on the LM-O dataset (i.e., cat, driller, and duck). Since the challenging LM-O dataset is typically used solely for evaluation, we directly use the same weights for DVMNet and 3DAHV that were trained in the LineMOD experiments.

Since the results on the rigorous metrics of `Acc@5/10°` are not reported in the 3DAHV [78] and DVMNet [77] paper, we thus retrain them using their official codes for the `Acc@5/10°` evaluation.

Moreover, as a label/training-free method, the performance of our method can be assessed on all the objects of LineMOD, YCB-V, and LM-O datasets, without the need to leave out any training data or leverage any external dataset. We report the performance of our method on the complete LineMOD, YCB-V, and LM-O datasets in Tables S5, S6, and S7 of the supplementary material.

In-dataset and Cross-dataset Evaluation. Beyond the unseen objects generalization, we also test the dataset-level generalization for the network-based methods including the state-of-the-art DVMNet [77] and 3DAHV [78], reporting both the *in-dataset* and the *cross-dataset* performance. In short, *in-dataset* and *cross-dataset* differ in whether the network needs to be finetuned on a subset that comes from the same dataset with the testing set (though not including the testing objects). Therefore, a good *cross-dataset* performance is desirable, as the network only needs to be (pre-) trained once on a large-scale external dataset without finetuning.

Specifically, for the *in-dataset* experiments, we follow the exact training protocols of DVMNet [77] and 3DAHV [78], which first pretrain on an external large-scale dataset Objaverse [10] then finetune on a certain dataset (e.g., LineMOD or YCB-V). For *cross-dataset* experiments, we use the pretrained weights from Objaverse directly without finetuning.

Note that our method, ZSP [16], and LoFTR [58] do not require a finetuning phase before evaluation, suggesting that our method, ZSP, and LoFTR naturally generalize to an arbitrary dataset[5].

Reference-Query Pair Generation. We follow DVMNet [77] and 3DAHV [78] to generate the reference-query pairs with sufficient overlaps for training and testing. Specifically, given a reference rotation R_r and a query rotation R_q, we first convert the rotation matrices R_r and R_q to Euler angles $(\alpha_r, \beta_r, \gamma_r)$ and $(\alpha_q, \beta_q, \gamma_q)$. Since the in-plane rotation γ does not influence the overlaps between the reference and query pair, it is set to 0 and converted back to the rotation matrix, i.e., $\tilde{R} = h(\alpha, \beta, 0)$ with h being Euler-angle to rotation matrix transformation. The overlap between the query and the reference is measured by the geodesic distance (i.e.,

[5]This is achieved by that i) the pose estimation phase of our method, ZSP, and LoFTR are general and do not involve learning a network, and ii) they all use generalizable feature extractors, i.e., DINOv2 or LoFTR

TABLE II
EXPERIMENTAL RESULTS ON LINEMOD. WE ILLUSTRATE BOTH THE EXPERIMENTAL SETTINGS AND THE PERFORMANCE. IN THE **RGB-D** CATEGORY, **BOTH** MEANS REQUIRING RGB-D IMAGE FOR BOTH QUERY AND REFERENCE. `Acc@`t° MEASURES THE PERCENTAGE OF THE ESTIMATED POSE WITHIN t° W.R.T. THE GROUND-TRUTH.

Method	Settings			Error↓	Acc @ t° (%) ↑			
	Training	Label	RGB-D	Mean Err	30°	15°	10°	5°
ZSP	✘	label-free	both	102.33	8.20	2.22	0.90	0.18
LoFTR	✓	pose+depth	no	63.88	23.94	10.80	6.82	2.42
RelPose++	✓	pose	no	46.60	42.50	15.80	–	–
3DAHV (cross-dataset)	✓	pose	no	69.24	21.20	5.52	2.52	0.44
3DAHV (in-dataset)	✓	pose	no	42.77	59.16	25.92	11.36	2.16
DVMNet (cross-dataset)	✓	pose	no	47.47	36.44	13.14	5.92	1.08
DVMNet (in-dataset)	✓	pose	no	33.28	55.02	22.38	10.66	2.72
Ours (init. only)	✘	label-free	reference	32.24	70.88	48.28	29.76	6.66
Ours (init. + refine)	✘	label-free	reference	**29.93**	**72.06**	**54.90**	**42.74**	**24.32**

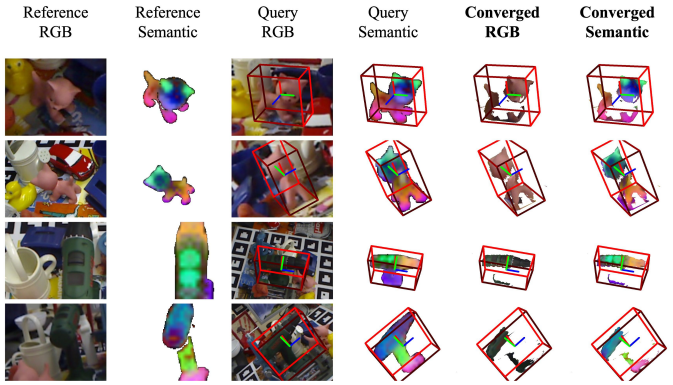

Fig. 5. **Qualitative results on LineMOD.** This figure shows that our method can handle partially occluded and texture-less objects. We use a 3D bbox to denote poses.

the pose difference in degree) between their in-plane-omitted rotation matrices $\tilde{R}_q$ and $\tilde{R}_r$ using Eq. (6). Finally, following DVMNet [77] and 3DAHV [78], we select the sampled pairs with $\tilde{D}$ less than 90°.

Following DVMNet [77] and 3DAHV [78], for each object, we generate 1000 pairs for testing, and 20000 pairs for training DVMNet, 3DAHV, and RelPose++. Fig. 4 illustrates the histograms depicting the statistics of the pairwise pose difference (geodesic distance between rotation matrices R_r and R_q) on the three datasets. All the experiments are carried out on the same testing reference-query pairs.

Implementation Details. For semantic feature extraction, we employ the output tokens from the last layer of the DINOv2 ViT-L model [46]. We use nvdiffrast [27] as our differentiable renderer. We uniformly sample $m = 200$ viewpoints and $n = 20$ in-plane rotations (resulting in 4000 initialization candidates), the maximal iteration number for differentiable rendering is set to $N = 30$. To backpropagate the refinement losses, we use an Adam optimizer [24] of 0.01 initial learning rate and decay by a `ReduceLROnPlateau` scheduler. All the experiments are conducted on a single NVIDIA 4090 GPU.

B. Experimental Results on the LineMOD Dataset

The results on the LineMOD dataset are illustrated in Table II. We paste the performances of RelPose++ from the 3DAHV paper [78]. We leave the `Acc@5/10°` performance of RelPose++ blank as those were not reported in [78] and the (pre-) training code of RelPose++ on the external large-scale Objaverse dataset is not available. Table II shows that our *label and training-free* method significantly outperforms the *supervised* state-of-the-art DVMNet w.r.t. all the metrics. In addition, the experimental results show that the performance of DVMNet and 3DAHV decreases when facing cross-dataset scenarios. In contrast, without training a network, our approach inherently generalizes across diverse datasets. Especially, our method significantly outperforms DVMNet (in-dataset) for 21.6% and 32.08% w.r.t. the rigorous `Acc@5/10°`. The qualitative results of our method are shown in Fig. 5, and comparisons with different methods are presented in Fig. S3 of the supplementary material. Our results on all the LineMOD objects are detailed in Table S5 of the supplementary material.

C. Experimental Results on the YCB-V Dataset

To compare with the state-of-the-art DVMNet [77], 3DAHV [78] and RelPose++ [34], we follow the protocols discussed in Sect. IV-A (In-dataset and Cross-dataset Evaluation) to obtain the in-dataset and cross-dataset performance of DVMNet [77] and 3DAHV [78], while RelPose++ is trained on the YCB-V dataset only. The performance on the YCB-V dataset is reported in Table III, where our method exhibits a significant improvement of 11.02% and 17.83% w.r.t. the state-of-the-art DVMNet (in-dataset), respectively on the challenging `Acc@5/10°` metrics. We showcase the qualitative results of our method on the YCB-V dataset in Fig. 6, and those across different methods can be found in Fig. S5 of the supplementary material. Our results on all the YCB-V objects are shown in Table S7 of the supplementary material.

D. Experimental Results on the LM-O Dataset

Finally, we carry out the experiments on the challenging LM-O Dataset with severe occlusions. Following DVMNet [77], we conduct the experiments on three unseen objects of the LM-O dataset, i.e., cat, driller, and duck. We note that the LM-O dataset is typically used solely for evaluation. Therefore, the results of DVMNet and 3DAHV are evaluated utilizing the weights finetuned on LineMOD. Nevertheless, since the weights of RelPose++ for the LineMOD dataset have not been released yet and LM-O (with only 8 objects) cannot provide sufficient leave-out data to train RelPose++, we thus do not include RelPose++ for comparison. The results from Table IV demonstrate the promising performance of our

TABLE III
EXPERIMENTAL RESULTS ON YCB-V. THE PERFORMANCE OF DVMNET, 3DAHV, AND RELPOSE++ IS OBTAINED BY TRAINING ON A LEAVE-OUT SUBSET OF 13 OBJECTS. OTHER PARAMETERS/SYMBOLS ARE THE SAME AS THOSE IN TABLE II.

Method	Settings			Error↓	Acc @ $t°$ (%) ↑			
	Training	Label	RGB-D	Mean Err	30°	15°	10°	5°
ZSP	✗	label-free	both	88.65	15.63	5.82	2.89	0.65
LoFTR	✓	pose+depth	no	68.65	29.45	13.56	7.9	3.19
RelPose++	✓	pose	no	57.41	23.60	7.13	3.28	0.76
3DAHV (cross-dataset)	✓	pose	no	66.61	35.06	16.18	8.28	1.50
3DAHV (in-dataset)	✓	pose	no	69.48	44.54	28.41	16.29	3.59
DVMNet (cross-dataset)	✓	pose	no	54.12	41.28	17.11	9.35	2.53
DVMNet (in-dataset)	✓	pose	no	48.88	51.71	27.04	14.03	3.16
Ours (init. only)	✗	label-free	reference	48.65	56.59	35.68	21.86	5.36
Ours (init. + refine)	✗	label-free	reference	**47.09**	**56.63**	**42.69**	**31.86**	**14.18**

TABLE IV
EXPERIMENTAL RESULTS ON LM-O. LM-O IS TYPICALLY USED SOLELY FOR TESTING WITH ONLY 8 OBJECTS UNDER SEVERE OCCLUSIONS. THE RESULTS OF DVMNET AND 3DAHV ARE TESTED DIRECTLY USING THE WEIGHTS TRAINED ON LINEMOD. SINCE THE MODEL WEIGHTS OF RELPOSE++ USED IN TABLE II WERE NOT RELEASED, WE DO NOT COMPARE OUR METHOD WITH RELPOSE++ IN THIS EXPERIMENT. OTHER PARAMETERS/SYMBOLS ARE THE SAME AS THOSE IN TABLE II.

Method	Settings			Error↓	Acc @$t°$ (%) ↑			
	Training	Label	RGB-D	Mean Err	30°	15°	10°	5°
ZSP	✗	label-free	both	103.70	7.10	1.67	0.60	0.07
LoFTR	✓	pose+depth	no	68.15	20.63	9.00	4.87	1.87
3DAHV (cross-dataset)	✓	pose	no	55.05	32.83	9.47	4.40	0.53
3DAHV (in-dataset)	✓	pose	no	62.30	40.29	10.57	3.84	0.57
DVMNet (cross-dataset)	✓	pose	no	51.75	35.52	12.94	5.30	1.33
DVMNet (in-dataset)	✓	pose	no	**48.55**	38.62	14.14	7.37	1.87
Ours (init. only)	✗	label-free	reference	55.94	53.80	31.72	17.18	2.80
Ours (init. + refine)	✗	label-free	reference	55.09	**54.50**	**34.97**	**23.00**	**6.83**

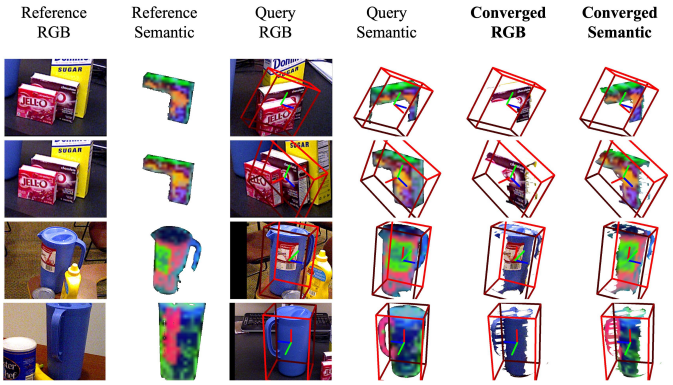

Fig. 6. **Qualitative results on YCB-V.** This figure shows that our method can handle partially occluded and texture-less objects. We use a 3D bbox to denote poses.

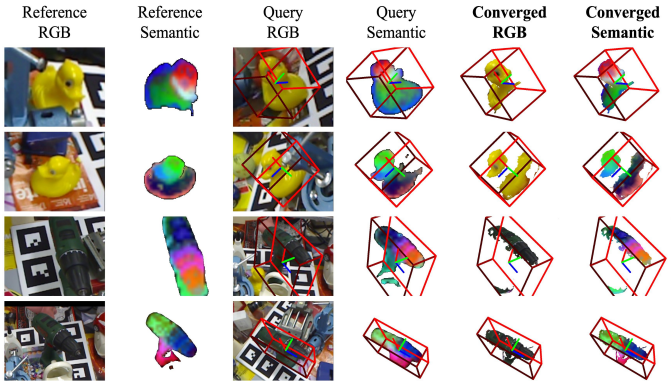

Fig. 7. **Qualitative results on LM-O.** This figure shows that our method can handle severely occluded and texture-less objects. We use a 3D bbox to denote poses.

method on the severely occluded LM-O dataset. We showcase our performance on the LM-O dataset in Fig. 7, and those across different methods are illustrated in Fig. S4 of the supplementary material. Our results on all the LM-O objects can be found in Table S6 of the supplementary material.

We observe that our results in terms of *Mean Err* are inferior to the in-dataset results of the state-of-the-art DVMNet and 3DAHV (though our method exhibits better Acc@$t°$ results). This can be attributed to the extensive occlusions presented in the LM-O dataset, which lead to numerous testing pairs lacking adequate overlap. Consequently, those testing pairs are difficult to handle by all the methods (and also challenging for humans). We show those samples as failure cases in Fig. 11 of Sect. V-F, as well as investigating the angle error distribution (ranging from 0 to 180 degrees) on the LM-O dataset in Fig. S1 of the supplementary materials. The statistics reveal that at lower angle error thresholds (e.g., for $t \leq 10, 20$ in Acc@$t°$), our approach substantially outperforms both DVMNet and 3DAHV. This indicates that for test pairs with sufficient overlaps (i.e., match-able testing pairs), our method delivers superior performance compared to the state-of-the-art DVMNet and 3DAHV.

V. ABLATION ANALYSIS

We carefully investigate the following issues by ablation: i) the contribution of each comprising element of our method, including *the back-surface culling*, and the usage of *RGB* or *semantic* modality in Sect. V-A; ii) the ablations on different semantic features in Sect. V-B; iii) the effects of different initialization strategies in Sect. V-C; iv) the effects of different

TABLE V
THE CONTRIBUTIONS OF **THE PROPOSED COMPRISING ELEMENTS** ON THE LINEMOD DATASET.

Metrics	Mean Err↓	Acc @30°↑	Acc @15°↑	Acc @10°↑	Acc @5°↑
w/o culling	38.09	67.46	52.32	40.82	23.58
only RGB	36.26	67.42	50.40	37.70	19.62
only semantic	31.31	69.32	50.86	38.80	19.22
Ours	**29.93**	**72.06**	**54.90**	**42.74**	**24.32**

TABLE VI
ABLATION ANALYSIS OF DIFFERENT **SEMANTIC FEATURES** ON THE LINEMOD DATASET.

Metrics	Mean Err↓	Acc @30°↑	Acc @15°↑	Acc @10°↑	Acc @5°↑
RGB	36.26	67.42	50.40	37.70	19.62
LoFTR	54.23	46.62	25.00	15.26	5.24
RGB + LoFTR	39.63	64.30	45.14	32.42	14.80
SD	38.45	57.94	37.28	26.32	12.12
RGB + SD	33.78	65.72	47.56	36.76	19.90
DINOv2	31.31	69.32	50.86	38.80	19.22
RGB + DINOv2	**29.93**	**72.06**	**54.90**	**42.74**	**24.32**

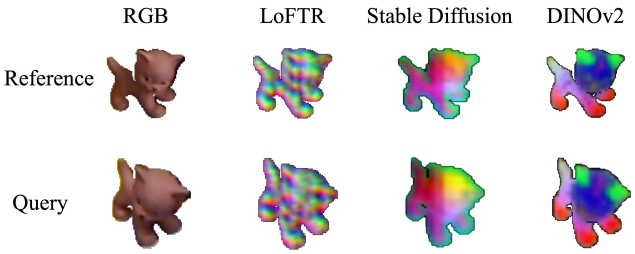

Fig. 8. Illustration of different semantic features as texture maps.

refinement iterations in Sect. V-D; v) the inference time statistics of our method and comparison with other baselines in Sect. V-E; and vi) the failure cases illustrations from the LM-O dataset in Sect. V-F.

A. The Contributions of the Proposed Comprising Elements

Despite the simplicity of our method, we are interested in investigating the influences for each of our comprising elements, namely *the back-surface culling*, and the usage of *RGB* or *semantic* modality. We perform those ablations on the LineMOD, and the results are reported in Table V.

As expected, removing each of our comprising elements results in a decreased performance, because all of them are exploited with clear motivations. Nonetheless, the encouraging observation is that our method is able to deliver promising results using only the **RGB** modality without the **semantic** map, which already outperforms the state-of-the-art **DVM-Net (in-dataset)** [77] in Table II across the rigorous Acc @5°, 10°, 15°, and 30°. This further illustrates the good applicability of our method when the pretrained DINOv2 model is not available.

B. Ablation on Semantic features

To further investigate the performance incorporating alternative semantic feature representations, we tested semantic features from LoFTR [58] and Stable Diffusion (SD) [53]. Table VI shows that i) LoFTR and SD features are inferior to DINOv2 for this task; ii) RGB-only performance surpasses the results based solely on LoFTR or SD features; iii) complementing LoFTR, SD, or DINOv2 with RGB improves the final performance.

Table VI reveals a significant performance gap among DINOv2, SD, and LoFTR features. To further investigate this, we visualize these features in Fig. 8, where the last three columns are dimensionality-reduced texture maps using PCA. Figure 8 shows that the DINOv2 feature maps best characterize the semantic cues, well complementing the RGB appearance, thus explaining its superior performance.

TABLE VII
EFFECTS OF DIFFERENT INITIALIZATION STRATEGIES USING THE LINEMOD DATASET.

Initial Strategy	Sampling Numbers	Error↓	Acc @ $t°$ (%) ↑			
		Mean Err	30°	15°	10°	5°
Random Init.	4000	31.73	70.90	52.66	40.88	23.08
Uniform Init.	4000	29.93	72.06	54.90	42.74	24.32

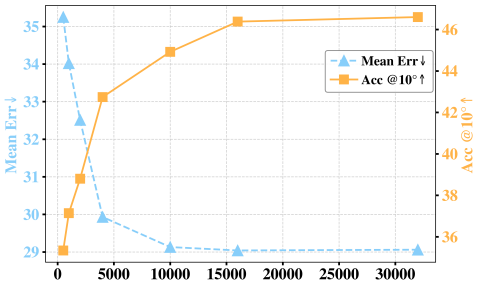

Fig. 9. **Ablation analysis on different number of uniformly initialized samples.** We tested sampling numbers of 500, 1000, 2000, 4000, 10000, 16000, 32000 on Mean Error and Acc @ 10°.

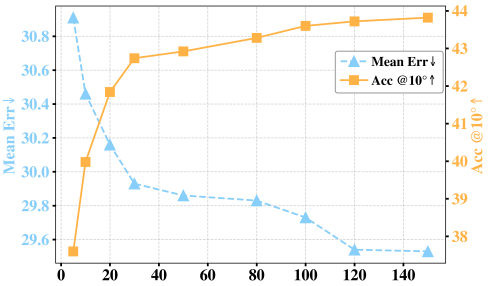

Fig. 10. **Performance trend w.r.t. number of refining iterations.** We tested iteration numbers of (5, 10, 20, 30, 50, 80, 100, 120, 150) on Mean Error and Acc @ 10°.

C. Effects of Different Initialization Strategies

The pose estimation performance under the render-and-compare paradigm is largely affected by the initialization [26, 31, 39, 44, 48, 71]. In the following, we investigate different initializations including: i) *random initialization*, where we randomly sample candidate poses and choose the best one; and ii) *uniform initialization*, where the candidate poses are uniformly sampled from a Fibonacci lattice with in-plane rotations [15], as detailed in Sect. III-D (Initialization). Table VII shows the performance of different initialization strategies using the LineMOD dataset, which demonstrates that the *uniform initialization* outperforms the *random initialization*.

Moreover, we also perform extensive ablations on the sampling densities using uniform initialization. As shown in Fig. 9, the performance boosts from 0 to 4000 samples, marginally improves from 4000 to 16000, and saturates after 16000 samples. In our experiments, we choose *uniform initialization with 4000 samples* to balance the performance and the efficiency.

D. Effects of Different Refinement Iterations.

Figure 10 illustrates the impact of the iteration numbers for our label/training-free refinement using the LineMOD dataset,

TABLE VIII
INFERENCE TIME STATISTICS OF OUR METHOD ON LINEMOD.

Semantic Fea. Extraction	Pose Initialization	Refinement	Total
0.11s	1.18s	1.02s	2.52s

TABLE IX
INFERENCE TIME COMPARISON ON LINEMOD.

Method	ZSP	RelPose++	LoFTR	3DAHV	DMVNet	Ours
Time	1.72s	0.69s	0.30s	0.04s	0.04s	2.52s

which demonstrates that the performance boosts from 0 to 30 iterations, marginally improves from 30 to 120, and saturates after 120 iterations. We thus set the iteration number to 30 in our main experiments to achieve a balance between performance and efficiency.

E. Analysis on the Inference Time

We collect the inference time per reference-query pair, averaged across the LineMOD datasets on a single 4090 GPU. We report the runtime for each stage of our method in Table VIII. Note that the initialization is efficient with much more candidate samples than the refinement, because those initializing candidate samples can be evaluated in parallel without backpropagation. Table VIII demonstrates the efficiency of our method with a per-pair runtime of 2.52 seconds in total.

We also present comparisons of our approach with the baseline methods, in terms of the inference time, in Table IX. While our method is slower in inference than the state-of-the-art feedforward models, our render-and-compare paradigm is training-free. This eliminates the numerous training hours required by the state-of-the-art feedforward methods, and ensures inherent generalization to unseen objects (i.e., training-free brings desirable training-data independence).

F. Illustrations of the Failure Cases

We show our failure cases on the LM-O dataset in Fig. 11, where there do not exist sufficient overlaps between the query and the reference. We note such an extremely degraded case as our limitation and discuss it in Sect. VI (Limitations and Future Works).

VI. DISCUSSIONS AND CONCLUSIONS

Limitations and Future Works. Our method has the following two limitations. Firstly, our method necessitates the depth information of the reference object as an input. To acquire the **metric** depth of the reference image, we evaluated a state-of-the-art monocular depth estimation algorithm [73]. The results and discussions in Sect. S3 of the supplementary shows that the metric depth estimation occasionally fails to generalize across different datasets, primarily stemming from the inherent metric scale ambiguities under varying camera parameters and diverse objects. This suggests that *the current limitations are likely to be overcome once a generalizable **metric** depth estimator becomes available*. Despite this, we note that depth sensors are commonly used in our primary application domain, i.e., robotics. Our empirical results, presented in Table S1 of the supplementary materials, demonstrate that our method

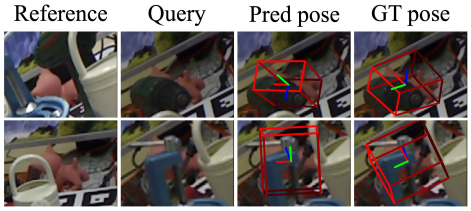

Fig. 11. **Failure cases of our method on the LM-O dataset**, where there do not exist sufficient overlaps between the query and the reference due to severe occlusions.

remains robust with imprecise depth obtained by a noisy depth sensor (simulated by adding noise to the ground-truth depth).

Secondly, our method is likely to fail in the severely degraded scenario where there do not exist adequate overlaps between the query and the reference (possibly caused by occlusions, e.g., Fig. 11). Future research with simultaneous render-and-compare and object completion (with minimal inconsistent hallucination) is a promising direction to explore.

We also note an additional future direction about adaptively determining the loss weights of the RGB pair and the semantic pair in Eq. (3) (preferably adapting in each refinement step), though we empirically showed that simply using equal weights (i.e., both set to 1) leads to promising results.

Conclusions. In this paper, we addressed the challenging generalizable relative pose estimation under a rigorous circumstance with only a single RGB-D reference and single RGB query pair as input, and the pose label is not a priori. We establish our label- and training-free method following the render-and-compare paradigm, by exploiting i) the 2.5D (i.e., RGB-D) rotatable reference mesh, ii) the semantic maps of both query and reference (extracted by a pretrained large vision model DINOv2), and iii) a differentiable renderer to produce and back-propagate losses to refine the relative pose. We carried out extensive experiments on the LineMOD, LM-O, and YCB-V datasets. The results demonstrate that our label/training-free approach surpasses the performance of state-of-the-art supervised methods, particularly excelling under the rigorous `Acc@5/10/15°` metrics.

REFERENCES

[1] P. Azad, T. Asfour, and R. Dillmann. Stereo-based 6D object localization for grasping with humanoid robot systems. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pages 919–924, 2007.
[2] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6D object pose estimation using 3D object coordinates. In *Proc. IEEE Eur. Conf. Comput. Vis.*, pages 536–551, 2014.
[3] Y. Bukschat and M. Vetter. EfficientPose: An efficient, accurate and scalable end-to-end 6D multi object pose estimation approach. *arXiv preprint arXiv:2011.04307*, 2020.
[4] A. Caraffa, D. Boscaini, A. Hamza, and F. Poiesi. Freeze: Training-free zero-shot 6d pose estimation with geometric and vision foundation models. In *Proc. IEEE Eur. Conf. Comput. Vis.*, pages 414–431, 2024.
[5] M. Caron et al. Emerging properties in self-supervised vision transformers. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 9650–9660, 2021.
[6] D. Chen, J. Li, Z. Wang, and K. Xu. Learning canonical shape space for category-level 6D object pose and size estimation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 11973–11982, 2020.
[7] J. Chen et al. Zeropose: Cad-prompted zero-shot object 6d pose estimation in cluttered scenes. *IEEE Transactions on Circuits and Systems for Video Technology*, 35:1251–1264, 2025.

[8] K. Chen and Q. Dou. SGPA: Structure-guided prior adaptation for category-level 6D object pose estimation. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 2773–2782, 2021.

[9] W. Chen, X. Jia, H. J. Chang, J. Duan, L. Shen, and A. Leonardis. FS-Net: Fast shape-based network for category-level 6D object pose estimation with decoupled rotation mechanism. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 1581–1590, 2021.

[10] M. Deitke et al. Objaverse: A universe of annotated 3D objects. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 13142–13153, 2023.

[11] Y. Di, F. Manhardt, G. Wang, X. Ji, N. Navab, and F. Tombari. SO-Pose: Exploiting self-occlusion for direct 6D pose estimation. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 12396–12405, 2021.

[12] Y. Gao and A. L. Yuille. Symmetric non-rigid structure from motion for category-specific object structure estimation. In *Proc. IEEE Eur. Conf. Comput. Vis.*, pages 408–424, 2016.

[13] Y. Gao and A. L. Yuille. Exploiting symmetry and/or manhattan properties for 3D object structure estimation from single and multiple images. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 7408–7417, 2017.

[14] Y. Gao and A. L. Yuille. Estimation of 3D category-specific object structure: Symmetry, manhattan and/or multiple images. *Int. J. Comput. Vis.*, 127:1501–1526, 2019.

[15] Á. González. Measurement of areas on a sphere using fibonacci and latitude-longitude lattices. *Mathematical Geosciences*, 42:49–64, 2010.

[16] W. Goodwin, S. Vaze, I. Havoutis, and I. Posner. Zero-shot category-level object pose estimation. In *Proc. IEEE Eur. Conf. Comput. Vis.*, pages 516–532, 2022.

[17] X. He et al. Keypoint-free one-shot object pose estimation without CAD models. In *Proc. Neural Inf. Process. Syst.*, volume 35, pages 35103–35115, 2022.

[18] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, and J. Sun. PVN3D: A deep point-wise 3D keypoints voting network for 6DoF pose estimation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 11632–11641, 2020.

[19] Y. He, Y. Wang, H. Fan, J. Sun, and Q. Chen. FS6D: Few-shot 6D pose estimation of novel objects. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 6814–6824, 2022.

[20] S. Hinterstoisser et al. Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In *Proc. Asian Conf. Comput. Vis.*, pages 548–562, 2012.

[21] T. Hodan, D. Barath, and J. Matas. EPOS: Estimating 6D pose of objects with symmetries. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 11703–11712, 2020.

[22] P. Kaushik, A. Mishra, A. Kortylewski, and A. Yuille. Source-free and image-only unsupervised domain adaptation for category level object pose estimation. In *Proc. Int. Conf. Learn. Representations*, 2024.

[23] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. SSD-6D: Making rgb-based 3D detection and 6D pose estimation great again. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 1521–1529, 2017.

[24] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proc. Int. Conf. Learn. Representations*, 2015.

[25] A. Kirillov et al. Segment anything. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 4015–4026, 2023.

[26] Y. Labbé et al. MegaPose: 6D pose estimation of novel objects via render & compare. In *Conference on Robot Learning*, pages 715–725, 2023.

[27] S. Laine, J. Hellsten, T. Karras, Y. Seol, J. Lehtinen, and T. Aila. Modular primitives for high-performance differentiable rendering. *Trans. Graph.*, 39(6):1–14, 2020.

[28] D.-T. Lee and B. J. Schachter. Two algorithms for constructing a delaunay triangulation. *Int. J. Comput. Inf. Sci.*, 9(3):219–242, 1980.

[29] T. Lee, B.-U. Lee, M. Kim, and I. S. Kweon. Category-level metric scale object shape and pose estimation. *IEEE Robot. Autom. Lett.*, 6(4):8575–8582, Oct. 2021.

[30] V. Lepetit, F. Moreno-Noguer, and P. Fua. EPnP: An accurate O(n) solution to the PnP problem. *Int. J. Comput. Vis.*, 81(2):155–166, 2009.

[31] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox. DeepIM: Deep iterative matching for 6D pose estimation. In *Proc. IEEE Eur. Conf. Comput. Vis.*, pages 683–698, 2018.

[32] Z. Li, G. Wang, and X. Ji. CDPN: Coordinates-based disentangled pose network for real-time RGB-based 6-DoF object pose estimation. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 7678–7687, 2019.

[33] R. Lian and H. Ling. CheckerPose: Progressive dense keypoint localization for object pose estimation with graph neural network. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 14022–14033, 2023.

[34] A. Lin, J. Y. Zhang, D. Ramanan, and S. Tulsiani. RelPose++: Recovering 6D poses from sparse-view observations. In *International Conference on 3D Vision*, pages 106–115, 2024.

[35] J. Lin, L. Liu, D. Lu, and K. Jia. SAM-6D: Segment anything model meets zero-shot 6D object pose estimation. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 27906–27916, 2024.

[36] J. Lin, Z. Wei, C. Ding, and K. Jia. Category-level 6D object pose and size estimation using self-supervised deep prior deformation networks. In *Proc. IEEE Eur. Conf. Comput. Vis.*, pages 19–34, 2022.

[37] J. Lin, Z. Wei, Z. Li, S. Xu, K. Jia, and Y. Li. DualPoseNet: Category-level 6D object pose and size estimation using dual pose network with refined learning of pose consistency. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 3560–3569, 2021.

[38] J. Lin, Z. Wei, Y. Zhang, and K. Jia. VI-Net: Boosting category-level 6D object pose estimation via learning decoupled rotations on the spherical representations. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 14001–14011, 2023.

[39] L. Lipson, Z. Teed, A. Goyal, and J. Deng. Coupled iterative refinement for 6D multi-object pose estimation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 6728–6737, 2022.

[40] Y. Liu et al. Gen6D: Generalizable model-free 6-DoF object pose estimation from RGB images. In *Proc. IEEE Eur. Conf. Comput. Vis.*, pages 298–315, 2022.

[41] F. Manhardt et al. CPS++: Improving class-level 6D pose and shape estimation from monocular images with self-supervised learning. *arXiv preprint arXiv:2003.05848*, 2020.

[42] V. N. Nguyen, T. Groueix, M. Salzmann, and V. Lepetit. GigaPose: Fast and robust novel object pose estimation via one correspondence. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 9903–9913, 2024.

[43] V. N. Nguyen, T. Y. Groueix, M. H. Salzmann, and V. Lepetit. Nope: Novel object pose estimation from a single image. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 17923–17932, 2024.

[44] V. N. Nguyen, Y. Hu, Y. Xiao, M. Salzmann, and V. Lepetit. Templates for 3D object pose estimation revisited: Generalization to new objects and robustness to occlusions. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 6771–6780, 2022.

[45] B. Okorn, Q. Gu, M. Hebert, and D. Held. Zephyr: Zero-shot pose hypothesis rating. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 14141–14148, 2021.

[46] M. Oquab et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research Journal*, pages 1–31, 2024.

[47] E. P. Örnek et al. FoundPose: Unseen object pose estimation with foundation features. In *Proc. Eur. Conf. Comput. Vis.*, pages 163–182, 2024.

[48] K. Park, A. Mousavian, Y. Xiang, and D. Fox. LatentFusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 10710–10719, 2020.

[49] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao. PVNet: Pixel-wise voting network for 6DoF pose estimation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 4561–4570, 2019.

[50] L. Pérez, Í. Rodríguez, N. Rodríguez, R. Usamentiaga, and D. F. García. Robot guidance using machine vision techniques in industrial environments: A comparative review. *Sensors*, 16(3):335, 2016.

[51] M. Rad and V. Lepetit. BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 3828–3836, 2017.

[52] T. Ren et al. Grounded SAM: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.

[53] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 10684–10695, 2022.

[54] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 4938–4947, 2020.

[55] I. Shugurov, F. Li, B. Busam, and S. Ilic. OSOP: A multi-stage one shot object pose estimation framework. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 6835–6844, 2022.

[56] Y. Su et al. ZebraPose: Coarse to fine surface encoding for 6DoF object pose estimation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 6738–6748, 2022.

[57] J. Sun et al. OnePose: One-shot object pose estimation without CAD models. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 6825–6834, 2022.

[58] J. Sun, Z. Shen, Y. Wang, H. Bao, and X. Zhou. LOFTR: Detector-free

local feature matching with transformers. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 8922–8931, 2021.
[59] D. J. Tan, F. Tombari, and N. Navab. Real-time accurate 3D head tracking and pose estimation with consumer rgb-d cameras. *Int. J. Comput. Vis.*, 126:158–183, 2018.
[60] B. Tekin, S. N. Sinha, and P. Fua. Real-time seamless single shot 6D object pose prediction. In *Proc. IEEE Conf. Comput, Vis. Pattern Recognit.*, pages 292–301, 2018.
[61] M. Tian, M. H. Ang, and G. H. Lee. Shape prior deformation for categorical 6D object pose and size estimation. In *Proc. IEEE Eur, Conf. Comput. Vis.*, pages 530–546, 2020.
[62] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(4):376–380, Apr. 1991.
[63] B. Wan, Y. Shi, and K. Xu. SOCS: Semantically-aware object coordinate space for category-level 6D object pose estimation under large shape variations. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 14065–14074, 2023.
[64] A. Wang, A. Kortylewski, and A. Yuille. Nemo: Neural mesh models of contrastive features for robust 3D pose estimation. In *Proc. Int. Conf. Learn. Representations*, 2021.
[65] G. Wang, F. Manhardt, F. Tombari, and X. Ji. GDR-Net: Geometry-guided direct regression network for monocular 6D object pose estimation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 16611–16621, 2021.
[66] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas. Normalized object coordinate space for category-level 6D object pose and size estimation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 2642–2651, 2019.
[67] T. Wang, G. Hu, and H. Wang. Object pose estimation via the aggregation of diffusion features. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 10238–10247, 2024.
[68] B. Wen et al. BundleSDF: Neural 6-DoF tracking and 3D reconstruction of unknown objects. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 606–617, 2023.
[69] B. Wen, W. Yang, J. Kautz, and S. Birchfield. FoundationPose: Unified 6D pose estimation and tracking of novel objects. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 17868–17879, 2024.
[70] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. *Robotics: Science and Systems*, 14(19), June 2018.
[71] Y. Xu, K.-Y. Lin, G. Zhang, X. Wang, and H. Li. RNNPose: Recurrent 6-DoF object pose refinement with robust correspondence field estimation and pose optimization. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 14880–14890, 2022.
[72] H. Yang and M. Pavone. Object pose estimation with statistical guarantees: Conformal keypoint detection and geometric uncertainty propagation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 8947–8958, 2023.
[73] L. Yang et al. Depth Anything v2. In *Proc. Neural Inf. Process. Syst.*, volume 37, pages 21875–21911, 2024.
[74] S. Zakharov, I. Shugurov, and S. Ilic. DPOD: 6D pose object detector and refiner. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, pages 1941–1950, 2019.
[75] H. Zhang and K. E. Hoff III. Fast backface culling using normal masks. In *Proc. 1997 Symp. Interactive 3D Graph.*, pages 103–106, 1997.
[76] J. Y. Zhang, D. Ramanan, and S. Tulsiani. RelPose: Predicting probabilistic relative rotation for single objects in the wild. In *Proc. IEEE Eur. Conf. Comput. Vis.*, pages 592–611, 2022.
[77] C. Zhao, T. Zhang, Z. Dang, and M. Salzmann. DVMNet: Computing relative pose for unseen objects beyond hypotheses. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 20485–20495, 2024.
[78] C. Zhao, T. Zhang, and M. Salzmann. 3D-Aware hypothesis & verification for generalizable relative object pose estimation. In *Proc. Int. Conf. Learn. Representations*, 2024.
[79] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for image restoration with neural networks. *IEEE Trans. Comput. Imag.*, 3(1):47–57, Mar. 2017.
[80] X. Zhao et al. Fast segment anything. *arXiv preprint arXiv:2306.12156*, 2023.

Yuan Gao (Member, IEEE) received the B.S. degree and the M.S. degree from Huazhong University of Science and Technology, and the Ph.D. degree from City University of Hong Kong, in 2009, 2012, and 2016, respectively. He was a visiting graduate researcher with University of California, Los Angeles in 2015, and a senior research scientist with Tencent AI Lab from 2017 to 2020. Currently, he is an Associate Professor with School of Artificial Intelligence, Wuhan University. His research interests include 3D computer vision, multi-task/modal learning, and efficient deep learning.

Yajing Luo received the B.S. degree from the School of Computer Science, Wuhan University in 2022. She is currently pursuing the Ph.D. degree with the School of Computer Science, Wuhan University. Her research interests include object pose estimation and 3D scene understanding.

Junhong Wang received the B.S. and M.S. degrees from Huazhong University of Science and Technology, Wuhan, China, in 2009 and 2012. He is now a graphics software engineer in Tencent Games since 2012. His research interests include 3D computer graphics and mobile rendering.

Kui Jia (Member, IEEE) received the B.E. degree from Northwestern Polytechnic University, Xi'an, China, in 2001, the M.E. degree from the National University of Singapore, Singapore, in 2004, and the Ph.D. degree in computer science from the Queen Mary University of London, London, U.K., in 2007. He was with the Shenzhen Institute of Advanced Technology of the Chinese Academy of Sciences, Shenzhen, China, Chinese University of Hong Kong, Hong Kong, the Institute of Advanced Studies, University of Illinois at Urbana-Champaign, Champaign, IL, USA, the University of Macau, Macau, China, South China University of Technology, Guangzhou, China. He is currently a Professor with the School of Data Science, the Chinese University of Hong Kong, Shenzhen, China. His recent research focuses on theoretical deep learning and its applications in vision and robotic problems, including deep learning of 3D data and deep transfer learning. He serves on the Editorial Boards of *IEEE Transactions on Image Processing*, and *Transactions on Machine Learning Research.*

Gui-Song Xia (Senior Member, IEEE) received the PhD degree in image processing and computer vision from CNRS LTCI, Télécom ParisTech, Paris, France, in 2011. From 2011 to 2012, he was a postdoctoral researcher with the Centre de Recherche en Mathématiques de la Decision, CNRS, Paris Dauphine University, Paris, for one and a half years. He is currently working as a full professor in computer vision and photogrammetry with Wuhan University. He has also been working as a visiting scholar at DMA, École Normale Supérieure (ENS-Paris) for two months, in 2018. He is also a guest professor of the Future Lab AI4EO in Technical University of Munich (TUM). His current research interests include mathematical modeling of images and videos, structure from motion, perceptual grouping, and remote sensing image understanding. He serves on the Editorial Boards of several journals, including *ISPRS Journal of Photogrammetry and Remote Sensing, Pattern Recognition, Signal Processing: Image Communications, EURASIP Journal on Image & Video Processing, Journal of Remote Sensing*, and *Frontiers in Computer Science: Computer Vision.*

Supplementary Material for the Paper: Towards Human-level 3D Relative Pose Estimation: Generalizable, Training-Free, with Single Reference

Yuan Gao*, Yajing Luo*, Junhong Wang, Kui Jia, Gui-Song Xia

We address the following issues in the supplementary material files:

1) Angle error distribution on LM-O in Sect. S1.
2) Experiments with imprecise input depth in Sect. S2.
3) Performance exploiting monocular *metric* depth estimation from advanced Depth Anything v2 [8] in Sect. S3.
4) Our results on the LineMOD [3], LM-O [1], and YCB-V [7] datasets w.r.t. **per object** in Sect. S4.
5) Qualitative Results on the LineMOD [3], LM-O [1], and YCB-V [7] datasets for all the methods in Sect. S5.
6) We also attached a video at https://www.youtube.com/watch?v=Ajr9ugjtoDo depicting the overview and the label/training-free refinement procedure of our method (i.e., the video version of Fig. 2 in the main text), as well as the qualitative results.

S1. Angle Error Distribution on the LM-O Dataset.

Figure S1 presents the angle error distribution (ranging from 0 to 180 degrees) for all the methods on the LM-O dataset. The statistics reveal that at lower angle error thresholds (e.g., for $t \leq 10, 20$ in Acc@$t°$), our approach substantially outperforms both 3DAHV [10] and DVMNet [9].

S2. Experiments with Imprecise Input Depth

As discussed in Sect. I.A (Applicability) in the main text, our method has the potential to use imprecise depth. We validate this on the LineMOD [3] dataset. Concretely, we simulate the imprecise depth by adding Gaussian noise $\mathcal{N}(0, \sigma)$ to the ground-truth depth map D_r, where σ is set to:

$$\sigma = \lambda * d, \qquad d = \max(D_r) - \min(D_r), \tag{1}$$

where d is the maximal depth difference of the input sample.

We validate different λ's with 0.001, 0.003, and 0.005 on the LineMOD [3] dataset, the results are shown in Table S1, which demonstrates that our method remains robust with imprecise depth obtained by a noisy depth sensor.

TABLE S1
Experiments with imprecise depth on LineMOD.

Metrics	Mean Err↓	Acc@30°↑	Acc@15°↑	Acc@10°↑	Acc@5°↑
$\lambda = 0.005$	33.10	69.52	52.66	40.16	20.64
$\lambda = 0.003$	30.92	71.44	54.96	42.90	23.10
$\lambda = 0.001$	30.11	72.00	55.10	43.04	24.22
Ours	**29.93**	**72.06**	**54.90**	**42.74**	**24.32**

S3. Performance exploiting monocular *metric* depth estimation

In order to further enhance our applicability without using reference depth as input, we further explored the advanced Depth Anything v2 (dpav2) [8] to estimate the monocular depth of our reference image.

To preserve the object shape, our method requires the ***metric*** depth, meaning the estimated depth z and spatial dimensions x, y should share the same unit of measurement (e.g., both in meters). This is in contrast to the ***relative*** depth, where the estimated depth and spatial appearance are subject to a scale ambiguity. Such an ambiguous scale distorts object shapes, e.g., given a centimeter spatial unit, an object could appear flattened if depth is measured in meters, or elongated if depth is in millimeters. Examples of scale-induced shape distortions are illustrated in Fig. S2.

In the following, we employ three configurations to obtain the metric depth from Depth Anything v2:

1) **Relative depth w/ Ground Truth align**: We use the relative depth estimated from the vanilla Depth Anything v2 model [8], and align its scale using the GT depth map. Denoted by *relative depth w/ GT align*, this approach yields the best results but is less practical as it requires GT depth annotations for scale alignment.
2) **In-dataset metric depth**: Following the procedure detailed in Section 7.3 of the Depth Anything v2 paper [8], we finetune the relative Depth Anything v2 model on the LineMOD [3] and YCB-V datasets [7], respectively, to obtain the corresponding metric depth models. Compared to *relative depth w/ GT align*, *in-dataset metric depth* is more practical but provides inferior results.
3) **Cross-dataset metric depth**: For zero-shot cross-dataset testing in our experiments, we employ the metric depth estimation model provided by Depth Anything v2, which was pretrained on the external Hypersim dataset [5]. This configuration *fully eliminates the requirement for in-dataset depth finetuning*. However, it yields limited accuracy, likely due to an imprecisely recovered depth

Y. Gao and G.-S. Xia are with the School of Artificial Intelligence, Wuhan University, Wuhan, China. E-mails: ethan.y.gao@gmail.com, guisong.xia@whu.edu.cn
Y. Luo is with the School of Computer Science, Wuhan University, Wuhan, China. E-mail: yajingluo@whu.edu.cn
J. Wang is with MoreFun Studio, Tencent Games, Tencent, Shenzhen, China. E-mail: junhongwang@tencent.com
K. Jia is with the School of Data Science, The Chinese University of Hong Kong, Shenzhen, China. E-mail: kuijia@cuhk.edu.cn
Corresponding authors: Yuan Gao, Gui-Song Xia.
* indicates equal contributions.

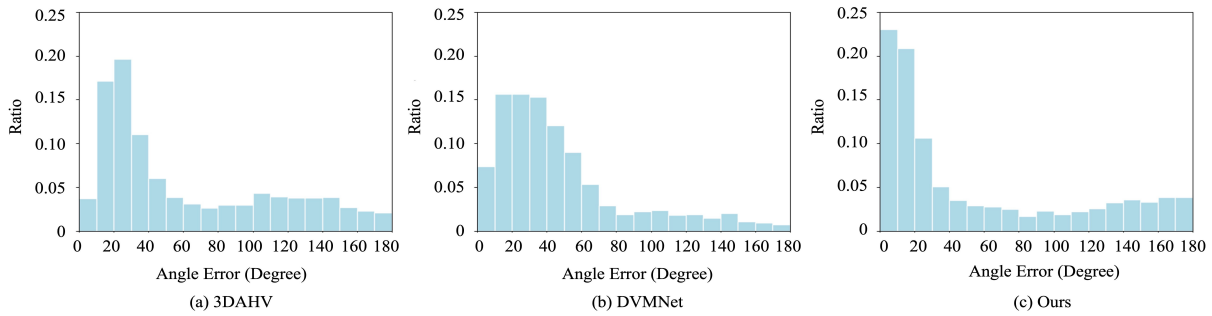

Fig. S1. **Angle Error Distribution on LM-O.**

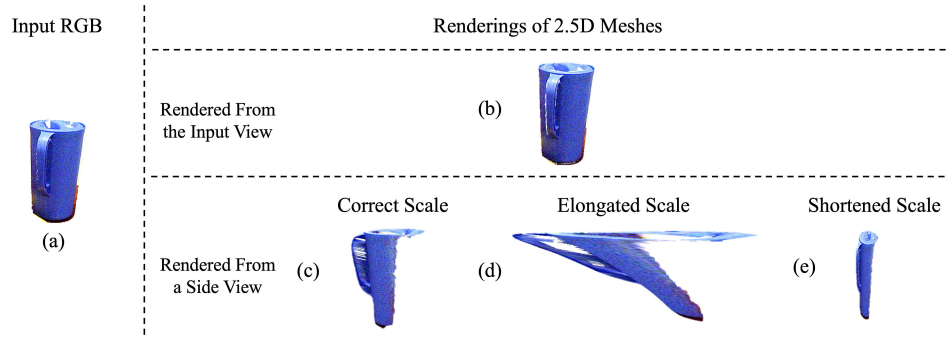

Fig. S2. **Illustration of scale-induced shape distortions.** Given an input image (a), the estimated depth z should align with the spatial dimensions x, y in scale, so as to obtain the correct shape (c). Incorrect depth scales will result in an elongated shape (d) or shortened shape (e). Subfigures (c), (d), and (e) are from a rotated side view of (b) for clearer illustrations. We thus require the ***metric*** depth **with a correct depth scale**, rather than the ***relative*** depth, to preserve the object shape.

scale caused by varying camera parameters and/or objects between the Hypersim pretraining set and zero-shot testing sets LineMOD [3], LM-O [1], and YCB-V [7].

The results shown in Tables S2 - S4 demonstrate that:

1) Ours, Ours (dpav2, relative depth w/ GT align), and Ours (dpav2, in-dataset metric depth) all outperform both SOTA DVMNet (in-dataset) and DVMNet (cross-dataset) [9] across the rigorous Acc @$5°, 10°, 15°$.
2) Our most applicable configuration, which is training-free and requires neither depth nor pose annotations, i.e., Ours (dpav2, cross-dataset metric depth), outperforms the SOTA DVMNet (cross-dataset) on the LineMOD dataset. However, it is inferior on YCB-V and LM-O, likely because that the heavy occlusions in these datasets lead to worse cross-dataset metric depth estimation.

Note that both the *relative depth w/ GT align* and *in-dataset metric finetune* approaches require additional depth labels to align or finetune a metric depth estimation model. On the other hand, the *cross-dataset metric depth* configuration performs suboptimally on some benchmarking datasets. Our current requirement for the reference depth is likely to be alleviated once a generalizable metric depth estimator becomes available.

TABLE S2
ABLATION OF PREDICTED DEPTH ON LINEMOD.

Method	Error↓	Acc @ $t°$ (%) ↑			
	Mean Err	30°	15°	10°	5°
DVMNet (cross-dataset)	47.47	36.44	13.14	5.92	1.08
DVMNet (in-dataset)	33.28	55.02	22.38	10.66	2.72
Ours (dpav2, relative depth w/ GT align)	32.04	70.34	48.98	34.84	15.32
Ours (dpav2, in-dataset metric depth)	41.54	59.80	34.18	21.7	8.38
Ours (dpav2, cross-dataset metric depth)	54.04	43.22	20.02	11.24	3.52
Ours	**29.93**	**72.06**	**54.90**	**42.74**	**24.32**

TABLE S3
ABLATION OF PREDICTED DEPTH ON LM-O.

Method	Error↓	Acc @$t°$ (%) ↑			
	Mean Err	30°	15°	10°	5°
DVMNet (cross-dataset)	51.75	35.52	12.94	5.30	1.33
DVMNet (in-dataset)	**48.55**	38.62	14.14	7.37	1.87
Ours (dpav2, relative depth w/ GT align)	59.28	47.70	28.78	18.31	5.50
Ours (dpav2, in-dataset metric depth)	66.58	41.56	22.58	13.11	2.53
Ours (dpav2, cross-dataset metric depth)	75.14	29.36	11.70	5.25	1.04
Ours	55.09	**54.50**	**34.97**	**23.00**	**6.83**

TABLE S4
ABLATION OF PREDICTED DEPTH ON YCB-V.

Method	Error↓	Acc @ $t°$ (%) ↑			
	Mean Err	30°	15°	10°	5°
DVMNet (cross-dataset)	54.12	41.28	17.11	9.35	2.53
DVMNet (in-dataset)	48.88	51.71	27.04	14.03	3.16
Ours (dpav2, relative depth w/ GT align)	54.54	53.21	40.19	29.44	13.41
Ours (dpav2, in-dataset metric depth)	57.15	49.10	33.71	23.64	10.05
Ours (dpav2, cross-dataset metric depth)	69.90	28.71	10.98	6.06	1.89
Ours	**47.09**	**56.63**	**42.69**	**31.86**	**14.18**

TABLE S5
OUR RESULTS ON THE LINEMOD DATASET W.R.T. PER OBJECT. THE OBJECTS WITH RED TEXT ARE THOSE USED FOR TESTING IN THE MAIN PAPER.

Object	Mean Err↓	Acc@30°↑	Acc@15°↑	Acc@10°↑	Acc@5°↑
ape	43.41	46.90	26.10	17.40	5.80
benchvise	17.79	87.30	75.60	64.60	42.10
camera	24.10	73.70	58.00	46.80	27.60
can	26.49	75.00	63.20	55.00	37.80
cat	33.90	68.00	52.20	40.20	22.00
driller	35.58	76.90	59.40	44.60	23.90
duck	38.30	54.40	29.30	17.50	6.00
eggbox	27.63	77.40	66.10	57.10	36.90
glue	46.35	55.30	38.20	28.00	13.70
holepuncher	26.25	76.50	64.30	52.30	26.80
iron	33.20	73.70	58.80	48.90	29.60
lamp	29.28	79.30	66.20	56.20	36.90
phone	25.82	80.80	64.60	50.20	27.30
average	31.39	71.17	55.54	44.52	25.88

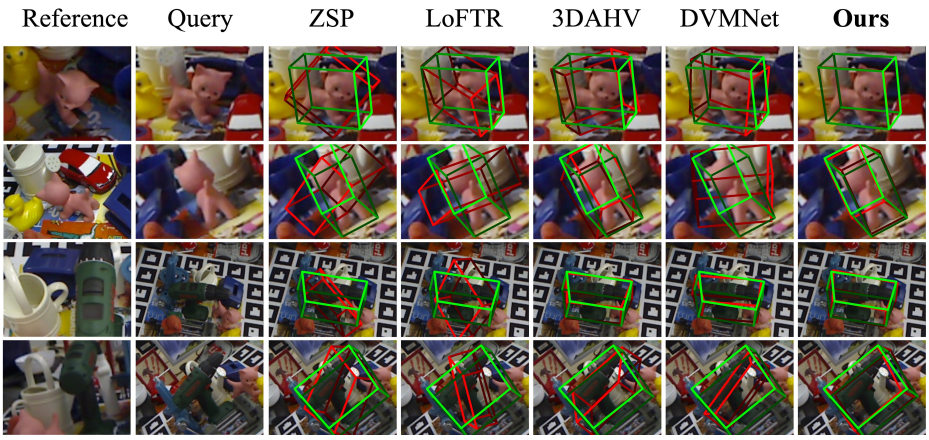

Fig. S3. **Qualitative results on LineMOD.** RelPose++ [4] did not release the LineMOD weights, the results of it in our main text were pasted from the 3DAHV paper [10], therefore the visualized results of RelPose++ are not included.

TABLE S6
OUR RESULTS ON THE LM-O DATASET W.R.T. PER OBJECT. THE OBJECTS WITH RED TEXT ARE THOSE USED FOR TESTING IN THE MAIN PAPER.

Object	Mean Err↓	Acc@30°↑	Acc@15°↑	Acc@10°↑	Acc@5°↑
ape	60.85	42.10	22.40	11.60	2.10
can	38.49	63.60	53.10	41.70	19.60
cat	55.84	53.70	33.00	22.30	8.60
driller	52.29	59.10	42.20	28.50	7.30
duck	57.13	50.70	29.70	18.20	4.60
eggbox	45.94	58.90	43.40	34.50	16.00
glue	61.52	46.50	30.70	19.20	9.20
holepuncher	42.51	62.20	47.20	33.00	10.40
average	51.82	54.59	37.72	26.13	9.73

S4. PER OBJECT RESULTS ON THE LINEMOD, LM-O, AND YCB-V DATASETS

We present our results w.r.t. per object of the full LineMOD [3], LM-O [1], and YCB-V [7] datasets in Tables S5, S6, and S7, respectively. The experimental settings are the same as those in the main text, i.e., Tables II-IV.

Tables S5, S6, and S7 show that our method performs well on all the objects of the three datasets without training, further validating the strong zero-shot unseen-object generalize-ability of our label/training-free method.

S5. QUALITATIVE RESULTS ON THE LINEMOD, LM-O AND YCB-V DATASETS

Qualitative results on the LineMOD [3], LM-O [1], and YCB-V [7] datasets are illustrated in Figs. S3, S4, and S5, respectively. The ground truth and predicted poses are visualized by axes and 3D bounding boxes.

As depicted in Figs. S3, S4, and S5, our method outperforms the state-of-the-art methods [2, 4, 6, 9, 10] qualitatively in all the three datasets.

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 *ECCV*, pages 536–551. Springer, 2014. 1, 2, 3

[2] Walter Goodwin, Sagar Vaze, Ioannis Havoutis, and Ingmar Posner. Zero-shot category-level object pose estimation. In *ECCV*, pages 516–532. Springer, 2022. 3

[3] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *ACCV*, pages 548–562. Springer, 2012. 1, 2, 3

[4] Amy Lin, Jason Y Zhang, Deva Ramanan, and Shubham Tulsiani. RelPose++: Recovering 6d poses from sparse-view observations. *arXiv preprint arXiv:2305.04926*, 2023. 3, 4

[5] Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *ICCV*, pages 10912–10922, 2021. 1

[6] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. In *CVPR*, pages 8922–8931, 2021. 3

[7] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017. 1, 2, 3

[8] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything v2. In *NeurIPS*, volume 37, pages 21875–21911, 2024. 1

[9] Chen Zhao, Tong Zhang, Zheng Dang, and Mathieu Salzmann. DVMNet: Computing relative pose for unseen objects beyond hypotheses. In *CVPR*, pages 20485–20495, 2024. 1, 2, 3

[10] Chen Zhao, Tong Zhang, and Mathieu Salzmann. 3d-aware hypothesis & verification for generalizable relative object pose estimation. In *ICLR*, 2024. 1, 3

TABLE S7
OUR RESULTS ON THE YCB-V DATASET W.R.T. PER OBJECT. THE OBJECTS WITH RED TEXT ARE THOSE USED FOR TESTING IN THE MAIN PAPER.

Object	Mean Err↓	Acc@30°↑	Acc@15°↑	Acc@10°↑	Acc@5°↑
002_master_chef_can	59.81	42.40	31.40	19.80	9.40
003_cracker_box	83.20	40.40	32.70	26.80	8.50
004_sugar_box	44.93	68.00	62.30	55.70	30.10
005_tomato_soup_can	61.65	36.50	25.80	19.70	9.40
006_mustard_bottle	20.76	86.40	83.30	77.80	59.70
007_tuna_fish_can	111.75	16.50	10.20	8.10	4.70
008_pudding_box	11.14	95.20	74.30	63.80	33.90
009_gelatin_box	8.13	99.50	87.20	78.30	32.40
010_potted_meat_can	99.01	26.80	22.30	13.20	4.60
011_banana	46.94	56.40	46.00	33.60	12.30
019_pitcher_base	33.43	58.60	41.40	28.50	9.50
021_bleach_cleanser	51.91	55.80	41.00	29.70	12.00
024_bowl	17.89	90.50	63.80	35.20	11.10
025_mug	43.98	40.80	22.90	15.50	3.30
035_power_drill	42.45	67.60	48.90	32.50	13.10
036_wood_block	39.52	69.10	48.50	30.50	10.50
037_scissors	27.55	83.50	60.00	41.10	15.40
040_large_marker	52.96	25.40	10.00	5.00	1.80
051_large_clamp	71.20	43.10	26.60	18.80	7.70
052_extra_large_clamp	88.61	27.00	14.90	7.40	2.80
061_foam_brick	27.89	81.90	73.00	50.10	13.30
average	49.74	57.69	44.12	32.91	14.55

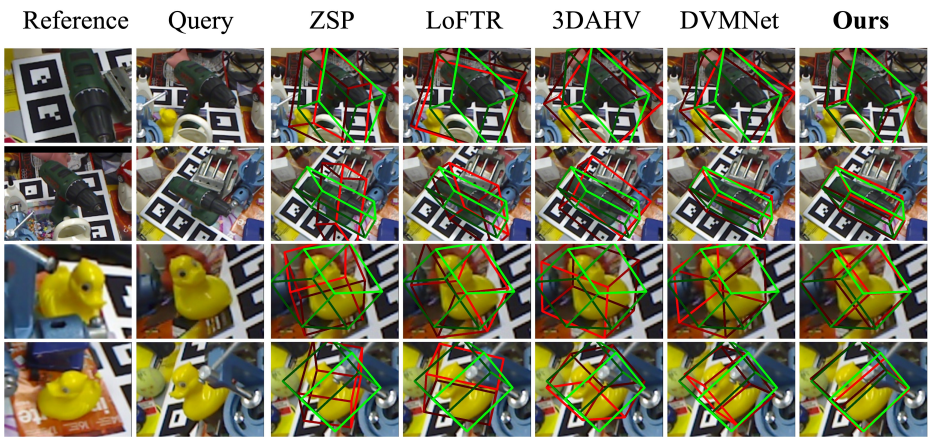

Fig. S4. **Qualitative results on LM-O.** LM-O is typically used solely to evaluate the models trained on LineMOD, since RelPose++ [4] have not released the LineMOD weights, the visualized results of RelPose++ are not included.

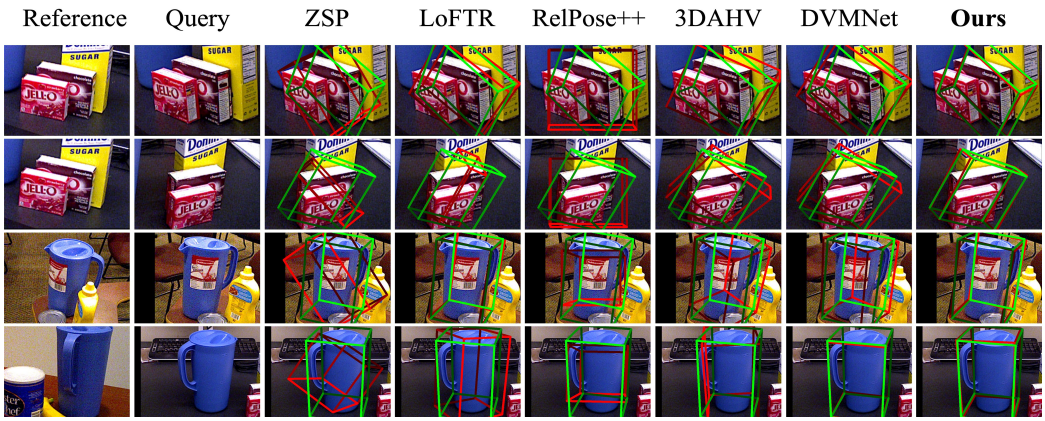

Fig. S5. **Qualitative results on YCB-V.** The predicted poses are visualized by red 3D bounding boxes while the ground truth poses are depicted by green 3D bounding boxes.