MFOS: Model-Free & One-Shot Object Pose Estimation

JongMin Lee[1], **Yohann Cabon**[2], **Romain Brégier**[2], **Sungjoo Yoo**[1], **Jerome Revaud**[2]

[1]Seoul National University
[2]Naver Labs Europe
sdrjseka96@naver.com, sungjoo.yoo@gmail.com, {romain.bregier, yohann.cabon, jerome.revaud}@naverlabs.com

Abstract

Existing learning-based methods for object pose estimation in RGB images are mostly model-specific or category based. They lack the capability to generalize to new object categories at test time, hence severely hindering their practicability and scalability. Notably, recent attempts have been made to solve this issue, but they still require accurate 3D data of the object surface at both train and test time. In this paper, we introduce a novel approach that can estimate in a single forward pass the pose of objects never seen during training, given minimum input. In contrast to existing state-of-the-art approaches, which rely on task-specific modules, our proposed model is entirely based on a transformer architecture, which can benefit from recently proposed 3D-geometry general pretraining. We conduct extensive experiments and report state-of-the-art one-shot performance on the challenging LINEMOD benchmark. Finally, extensive ablations allow us to determine good practices with this relatively new type of architecture in the field.

Introduction

Being able to estimate the pose of objects in an image is a mandatory requirement for any tasks involving some kind of interactions with objects. The past decade has seen a surge of research in 3D vision, with potential applications ranging from robotics (Hietanen et al. 2019; Deng et al. 2019) to VR/AR (Belghit et al. 2018; Marchand, Uchiyama, and Spindler 2016). These applications require pose estimators that are accurate, robust and scalable. In this context, we tackle the problem of object pose estimation from a single image, *i.e*. we aim at extracting the 6D pose of a target object relatively to the camera.

Object pose estimation is a long-studied research topic. Earlier approaches were holistic (Hinterstoisser et al. 2011, 2012a,b; Rios-Cabrera and Tuytelaars 2013; Kehl et al. 2016), based on sliding-window template-based matching (Song 2017; Henriques et al. 2014) or local features (Brachmann et al. 2014, 2016; Tejani et al. 2014). These methods were heavily handcrafted to exploit 3D priors. With the advent of deep learning, a new training-based paradigm emerged for object pose estimation (Xiang et al. 2017; Li, Wang, and Ji 2019; Wang et al. 2021; Park, Patten, and Vincze 2019), the idea of letting a deep network end-to-end predict the pose of an object from an image, given sufficient training data (images of the same object in various poses). While significantly improving in robustness and accuracy, these methods have the disadvantage of being *model-specific*: they can only cope with objects seen during training.

While some works have broadened the model scope to object categories rather than object instances (Wang et al. 2019; Tian, Jr., and Lee 2020; Lee et al. 2021; Chen, Li, and Xu 2020; Chen et al. 2020), the trained model is still only suitable for objects or categories seen during training. To remedy this shortcoming, recent learning-based methods that can generalize to objects unseen during training, denoted as *one-shot*, have been proposed. In practice however, they rely on 3D models (Cai, Heikkilä, and Rahtu 2022; Shugurov et al. 2022), video sequences (Wen and Bekris 2021) or additional depth maps (He et al. 2022) of the objects at test time. These requirements severely hinder the practicality and scalability of such approaches.

In this paper, we propose a novel approach to address the limitations of previous methods for object pose estimation. As illustrated in Figure 1, our method can estimate the pose of a target object from a single image, denoted as *query* image in the following. To specify the object of interest at inference, we provide as input the object's 2D bounding box, a rough estimate of the object size, and a small collection of *reference* images of the target object with known poses. These inputs can be obtained via scalable and straightforward methods, *e.g*. fiducial markers (AprilTags) (Olson 2011) or SfM (Schönberger and Frahm 2016). Similar to previous work, our model outputs a dense 2D-3D mapping from which the object pose can be obtained straightforwardly (Zakharov, Shugurov, and Ilic 2019; Li, Wang, and Ji 2019; Park, Patten, and Vincze 2019).

Our approach is entirely implemented using Vision Transformer (ViT) blocks (Dosovitskiy et al. 2021). Doing so enables us to leverage a powerful pretraining technique specifically tailored to 3D vision that can embed strong geometric priors in the network. Specifically, we initialize our network from an off-the-shelf model pretrained using Cross-View Completion (CroCo) (Weinzaepfel et al. 2022a). We show that this pretraining considerably boost the generalization capabilities of our method, making it possible to estimate the pose of target objects unseen during training. Inspired by BB8 (Rad and Lepetit 2017), we simply yet effectively

encode the object pose with a *proxy shape* positioned and scaled according to the object's pose and dimensions, respectively. We show that using rectangular cuboid as proxy shape works well in practice and allows us to deal with objects of unknown shape at test time. Our overall architecture is a generalization of the CroCo architecture (Weinzaepfel et al. 2022b) to multiple reference images (instead of just one in CroCo). It is computationally efficient at both training and test time, and it requires a single forward pass.

To ensure robust generalization of our model, we train it on a diverse set of object-centric data, including BOP challenge data (Hodan et al. 2018), OnePose (Sun et al. 2022) and the ABO datasets (Collins et al. 2022), which include a variety of objects along with their poses. Extensive ablation studies highlight the importance of mixing several data sources, and enable to validate our design choices for this relatively novel type of architecture in the field. Our method outperforms all existing one-shot pose estimation methods on the LINEMOD benchmark (Hinterstoisser et al. 2012b) and performs well in the OnePose benchmark (Sun et al. 2022). Finally, to demonstrate the robustness of our method in real-world scenarios, we present evaluation results in which a limited number of reference images are provided, outperforming all other methods.

In summary, we make several contributions. First, we propose a novel transformer-based architecture for object pose estimation that can handle unseen objects at test time without resorting to 3D models. Second, we demonstrate the importance of generic 3D-vision pretraining for better generalization in the context of object pose estimation. Third, we conduct extensive evaluations and ablations, and show that our method outperforms other existing one-shot methods on several benchmarks. In particular, our method does not significantly compromise performance in situations with limited information, such as a restricted number of reference images.

Related Work

Model-specific approaches are only able to estimate the pose of objects for which the method has been specifically trained. Some of these methods directly regress 6D pose from RGB images (Xiang et al. 2017; Li, Wang, and Ji 2019; Li and Ji 2020; Wang et al. 2021; Do et al. 2018), while others output 2D pixel to 3D point correspondences from which 6D pose can be solved using PnP (Park, Patten, and Vincze 2019; Chen et al. 2022; Peng et al. 2018; Zakharov, Shugurov, and Ilic 2019; Rad and Lepetit 2017). In this latter case, most methods leverage accurate 3D mesh models for each object as ground-truth for the 2D-3D mapping (Park, Patten, and Vincze 2019; Kehl et al. 2017; Iwase et al. 2021). Although high pose accuracy can be achieved this way, the requirement for exact mesh models hinders scalability and practical use in many application scenarios. To eliminate the need for 3D models, recent works (Park et al. 2019; Lin et al. 2020) use neural rendering models (Mildenhall et al. 2020) for pose estimation. Regardless, model-specific methods remains not scalable, as they need to be retrained for each new object.

Category-level methods learn the shared shape prior within a category and thus eliminate the need for instance-level mesh models at test time (Wang et al. 2019; Tian, Jr., and Lee 2020; Lee et al. 2021; Wang, Chen, and Dou 2021; Chen et al. 2020, 2021; Chen, Li, and Xu 2020; Chen and Dou 2021; Pavllo et al. 2023). Most of these approaches try to infer correspondences from pixels to 3D points in a Normalized Object Coordinate Space (NOCS). Nevertheless, category-level methods still face limitations. Namely, they can handle only a restricted number of categories and cannot handle objects from unknown categories.

Model-agnostic methods focus on estimating the poses of objects unseen during training, regardless of their category (Wen and Bekris 2021; He et al. 2022; Cai, Heikkilä, and Rahtu 2022; Gou et al. 2022; Shugurov et al. 2022; Liu et al. 2023; Sun et al. 2022; He et al. 2023). These methods assume that some additional input about the object at hand is provided at test time in order to define a reference pose (otherwise, the pose estimation problem would be ill-defined). BundleTrack (Wen and Bekris 2021) and Fs6D (He et al. 2022), for instance, requires RGB-D input sequences at inference time. More recently, several methods have been proposed for pose estimation of previously unseen objects, given their 3D mesh models. For instance, OVE6D (Cai, Heikkilä, and Rahtu 2022) utilizes a codebook to encode the 3D mesh model. OSOP (Shugurov et al. 2022) employs 2D-2D matching and PnP solving techniques based on the 3D mesh model of the object. However, these methods require dense depth information, video sequences or 3D meshes that can be challenging to obtain without sufficient time or specific devices.

One-shot image-only pose estimation methods are a subset of model-agnostic methods that only require minimal input at test time, *i.e*. a set of reference images with annotated poses (Liu et al. 2023; Sun et al. 2022; He et al. 2023). Gen6D (Liu et al. 2023) uses detection and retrieval to initialize the pose of a query image and then refines it by regressing the pose residual. However, it requires an accurate pose initialization and struggles with occlusion scenarios. OnePose (Sun et al. 2022) and OnePose++ (He et al. 2023) beforehand reconstruct the object 3D point-cloud from the set of reference images using COLMAP (Schönberger and Frahm 2016), from which 2D-3D correspondences are obtained. Although not requiring an explicit 3D mesh models, these two method still need to reconstruct a 3D point-cloud under the hood, which is complex, prone to failure, and not real-time. In comparison, our method do not need 3D mesh model nor reconstructed point-cloud to infer the object pose.

Method

In this section, we first describe the architecture of the proposed model-agnostic approach, then we describe its associated training loss. Afterwards, we present training details and the 6D pose inference procedure.

Model Architecture

Our model takes as input a query image $\mathcal{I}_q$ of the target object $\mathcal{O}$ for which we wish to estimate the pose, and a set of K reference images $\{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_K\}$ showing the same object under various viewpoints, for which the object pose is

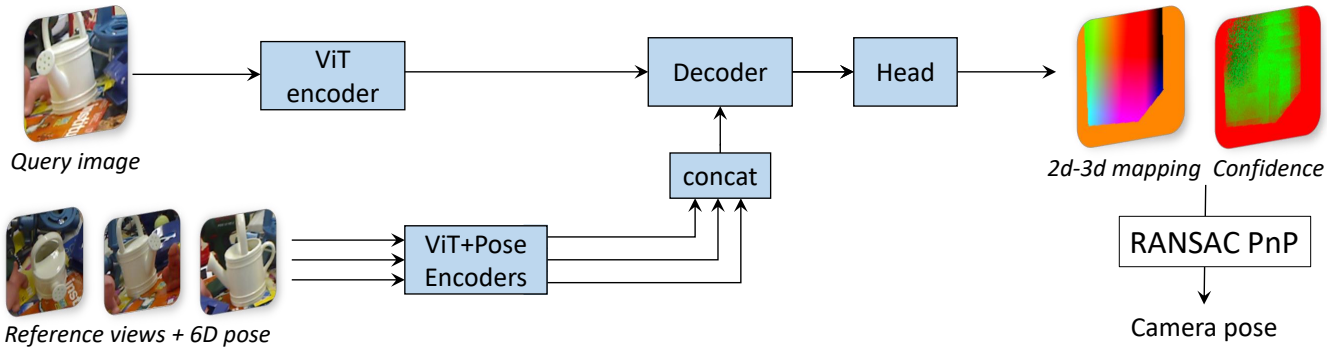

Figure 1: Overview of the method. Our model takes as input a query image and a set of K reference views of the same object seen under different viewpoints (annotated with pose information). We use a vision transformer (ViT) to first encode all images. Reference images are encoded with their corresponding object pose annotations. Then, a transformer decoder jointly processes features from the query and reference images. Finally, a prediction head outputs a dense 2D-3D mapping and a corresponding confidence map, from which we can recover the query object pose by solving a PnP problem.

known. We denote by $\mathbf{P}_i = \{(\mathbf{R}_i, \mathbf{t}_i)\}$ the pose of the object relatively to the camera in the reference image $\mathcal{I}_i$. Here we assume prior knowledge of the object instance in the query image, which is typically provided by an object detector or a retrieval system applied beforehand. For the sake of simplicity and without loss of generality, we also assume that all images (query and reference images) are approximately cropped to the object bounding box.

Overview of the architecture. Figure 1 shows an overview of the model architecture. First, the query and reference images are encoded into a set of token features with a Vision Transformer (ViT) encoder (Dosovitskiy et al. 2021). For each reference image, the object pose is encoded and injected into the image features using cross-attention. This latter module, to which we refer as *pose encoder*, outputs visual features *augmented* with 6D pose information. A transformer decoder then jointly process the information from the query features with the augmented reference features. Finally, a prediction head outputs 3D coordinates for each pixel of the query image, from which we recover the 6D pose in the query image.

Image encoder. We use a vision transformer (Dosovitskiy et al. 2021) to encode all query and database images. In practice, we use a ViT-Base model, *i.e.* 16×16 patches with 768-dimensional features, 12 heads and 12 blocks. Following (Xie et al. 2023; Weinzaepfel et al. 2022a), we use RoPE (Su et al. 2021) relative position embeddings. As a result of the ViT encoding, we obtain sets of token features denoted $\mathcal{F}_q$ for the query and $\mathcal{F}_i$ for the reference image $\mathcal{I}_i$ respectively:

$$\begin{cases} \mathcal{F}_q = \text{ImageEncoder}\left(\mathcal{I}_q\right), \\ \mathcal{F}_i = \text{ImageEncoder}\left(\mathcal{I}_i\right), i = 1 \dots K. \end{cases} \tag{1}$$

Pose encoder. There are multiple ways of inputting a 6D pose $\mathbf{P}_i$ to a deep network, see (Brégier 2021). We opt for an image-aligned pose representation which blends with the visual representation $\mathcal{F}_i$. Specifically, as shown in Figure 3, we transform the pose into an image by rendering 3D coordinates of a *proxy shape* (*e.g.* a cuboid or an ellipsoid), scaled according to the object dimension, and positioned according

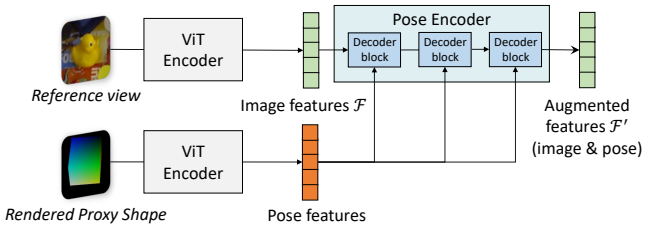

Figure 2: Architecture of the Pose Encoder. The pose encoder combines the reference image features $\mathcal{F}$ with the annotated object pose, in the form of a rendered 3D proxy shape, yielding the pose-augmented features $\mathcal{F}'$.

to the 6D pose. As illustrated in Figure 2, this 3-channel image is fed to another ViT, and then mixed with the visual features $\mathcal{F}_i$ through the cross-attention layers of transformer decoder, yielding the pose augmented features $\mathcal{F}'_i$:

$$\mathcal{F}'_i = \text{PoseEncoder}\left(\mathcal{F}_i, \text{ViT}(\text{Render}(\mathbf{P}_i))\right). \tag{2}$$

Decoder. The next step is to extract relevant information from the reference images with respect to the query image. To that aim, we again leverage a transformer decoder that compares the query features $\mathcal{F}_q$ to all concatenated tokens $\mathcal{F}'_i$ from the augmented reference images via cross-attention.

Prediction head. After obtaining the token features from the last transformer decoder block, we project them using a linear head and reshape the result as a 4-channel image with the same resolution as the query image. For each pixel, we thus predict the 3D coordinates of the associated point on the proxy shape, and an additional 4th channel yields the confidence τ (see below). Note that we predict the 3D coordinates on the surface of the proxy shape, not those on the surface of the target object. Finally, a robust PnP estimator extracts the pose from this predicted 2D-3D mapping.

Training Losses

3D regression loss. A straightforward way to train the network is, for each pixel i, to regress the ground-truth 3D coor-

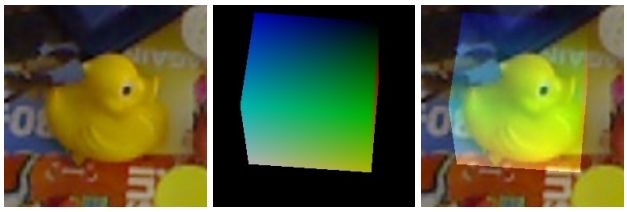

Figure 3: Reference view and its associated proxy shape. Illustration of a cuboid proxy shape used to jointly represent the object pose and dimensions. The proxy shape is rendered into a dense 3D coordinate map *w.r.t.* the object coordinate system, represented here as a 3-channel image, a data-format which is well suited for vision transformers.

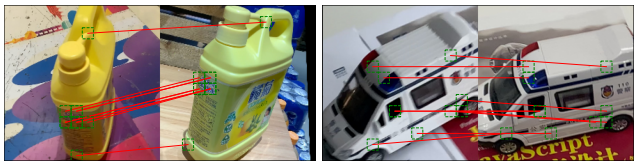

Figure 4: Visualization of the cross-attention in the decoder. Here we plot the top-10 attentions as correspondences between 2 tokens (*i.e.* 16x16 patches) in the query and reference image, respectively.

dinates of the proxy shape at this pixel. We use an Euclidean loss for pixels where such ground-truth is available:

$$\mathcal{L}^{(i)}_{\text{regr}} = \left\| \hat{\mathbf{y}}_i - \mathbf{y}^{gt}_i \right\|, \tag{3}$$

where $\hat{\mathbf{y}}_i \in \mathbb{R}^3$ is the network output for the i^{th} pixel of the query image, and $\mathbf{y}^{gt}_i$ is the corresponding ground-truth 3D point.

Pixelwise confidence. Since it is unlikely that all pixels can get correctly mapped during inference (*e.g.* background pixels), it is important to assert the likelihood of correctness of the predicted 2D-3D mapping for each pixel. Following Kendall, Gal, and Cipolla (2018), we therefore jointly predict a per-pixel indicator $\tau_i > 0$ to modulate the pixelwise loss $\mathcal{L}_{\text{regr}}$ to form the final loss as follows:

$$\mathcal{L}^{(i)}_{\text{final}} = \tau_i \mathcal{L}^{(i)}_{\text{regr}} - \log \tau_i. \tag{4}$$

Note that τ can be interpreted as the confidence of the prediction, as if τ_i is low for pixel i, the corresponding error $\mathcal{L}^{(i)}$ at this location will be down-weighted, and *vice versa*. For pixels outside the proxy shape, we set $\mathcal{L}^{(i)}_{\text{regr}} = E$, where E is a constant representing a large regression error. The second term of the loss acts as a regularizer, so as to prevent the model from getting under-confident everywhere.

Training Details

Training data. To ensure the generalization capability of our model, we train it on a diverse set of datasets covering a large panel of diversity. Specifically, we choose the large-scale ABO dataset (Collins et al. 2022), which comprises 580K images from 8,209 sequences, featuring 576 object categories (mostly furniture) from amazon.com. We also use for training some datasets of the BOP challenge (Hodan et al. 2018), namely T-LESS, HB, HOPE, YCB-V, RU-APC, TUD-L, TYO-L and ICMI. We exclude 3 objects from the HB dataset which exist in the LINEMOD dataset, in order to evaluate our generalization capabilities on this benchmark. In total, we consider 150K synthetic physically-based-rendered images and 53K real images, featuring 153 objects, from the BOP challenge for training. Additionally, we incorporate the OnePose dataset (Sun et al. 2022), which includes over 450 video sequences of 150 objects captured under various background environments.

We consider the convex hull of the 3D object mesh or its 3D bounding box to define the dimensions of the proxy shape (depending on which one is available).

Memory optimization. During training, we feed the network with batches of $16 \times 48 = 768$ images, each batch being composed of 16 objects for which 16 query and 32 reference images are provided (48 images in total). Since queries of the same object attend to the same set of reference images, we precompute features $\{\mathcal{F}'\}$ for these reference images and share them across all queries. Furthermore, by a careful reshaping of the tensors in-place in the query decoder, we can resort to vanilla attention mechanisms without any copy in memory (see Supplementary material for details). In addition to considerably reducing the memory requirements, this optimization significantly speeds up training.

Network architecture and training hyper-parameters. We use a ViT-Base/16 (Dosovitskiy et al. 2021) for the image encoder. The decoder is identical, except it has additional cross-attention modules. For the pose encoder, we use a single-layer ViT to encode the proxy shape rendering, and 4 transformer decoder blocks to inject the pose information into the visual representation. We use relative positional encoding (RoPE (Su et al. 2021)) for all multi-head attention modules. We train our network with AdamW with $\beta = (0.9, 0.95)$ and a cosine-decaying learning rate going from 10^{-4} to 10^{-6}. We initialize the network weights using CroCo v2 (Weinzaepfel et al. 2022a), a recently proposed pretraining method tailored to 3D vision and geometry learning.

Data augmentation. We recale and crop all images to a resolution of 224×224 around the object location. We apply standard augmentation techniques for cropping, such as random shifting, scaling and rotation to increase the diversity of our training data. We also apply augmentation to the input of our *pose encoder* to improve generalization. We specifically apply random geometric 3D transformations to the proxy shape pose and coordinates, including 3D rotation, translation and scaling. When choosing the set of 32 reference images for each object, we select 8 reference images at random across the entire pool of reference images for this object, and the remaining 24 views are selected using a greedy algorithm, *i.e.* farthest sampling that minimizes blind spots.

Inference Procedure

3D proxy shape. Our pose encoder receives multiple reference images of the target object and their corresponding

6D poses. Given a proxy shape template (*e.g*. a cuboid or an ellipsoid), we first align the 3D proxy shape centroid with the object center (according to the ground-truth pose). We then scale the proxy shape according to the target object dimensions. The generated 3D proxy shape is then transformed according to the object pose and rendered to the camera, yielding a 3D point map, see Figure 3.

Predicting object poses. To solve the object pose in a given query image, we sample K reference views among all the available reference views for this object. We use a greedy algorithm (Eldar et al. 1997) to maximize the diversity of viewpoints in the selected pool of views. From this input, our model predicts a dense 2D-3D mapping and an associated confidence map, as can be seen in Figure 1. We then filter out regions for which the confidence is below a threshold τ. Finally we use an off-the-shelf PnP solver to obtain the predicted object pose. Specifically, we rely on SQ-PnP (Terzakis and Lourakis 2020) with 1024 2D-3D correspondences, randomly sampled according to the confidence of the remaining points, a maximum of 1000 iterations and a reprojection error threshold of 5 pixels.

Experiments

Dataset and Metrics

Test benchmarks. We use the test splits of the training datasets explained earlier. In more details, we evaluate on the LINEMOD (Hinterstoisser et al. 2012b) dataset, a subset of the BOP benchmark (Collins et al. 2022), a widely-used dataset for object pose estimation comprising 13 models and 13K real images. For the evaluation, we use the standard train-test split proposed in (Li, Wang, and Ji 2019) and follow the protocol defined in OnePose++ (He et al. 2023), using their open-source code and detections from the off-the-shelf object detector YOLOv5 (Jocher et al. 2020). In more details, we keep approximately 180 real training images as references, discarding the 3D mesh models and only using the pose annotations, while all remaining test images are used for evaluation. For the Onepose (Sun et al. 2022) and ABO (Collins et al. 2022) datasets, we use the official test splits as well. We also use OnePose-LowTexture dataset (He et al. 2023), where there are 40 household low-textured objects for evaluation.

Metrics. We use the *cm-degree* metric to evaluate the accuracy of our predicted poses on both datasets. The rotation and translation errors are calculated separately, and a predicted pose is considered correct if both its rotation error and translation error are below a certain threshold. For the LINEMOD dataset, mesh models are available to evaluate the accuracy, and therefore, we employ two additional metrics: the *2D projection metric* and the *ADD metric*. We set the threshold for the *2D projection metric* to 5 pixels. To compute the *ADD metric*, we express coordinates of the 3D model's vertices in both the ground truth and predicted poses, and calculate the average distance between the two sets of transformed points. We consider a pose accurate if the average pointwise distance is smaller than 10% of the object's diameter. For symmetric objects, we consider the average point-to-set distance (*ADD-S*) instead (Xiang et al. 2017).

Ablative Study

We first conduct several ablative studies to measure the impact of critical components in our method, such as the choice of proxy shape, training data and pretraining, or the number of reference images. For these experiments, we report numbers on subsets of the three benchmarks mentioned above. We uniformly sample 5000 queries from ABO dataset and report each time a few adequate metrics. Unless specified otherwise, we use the same training sets, hyper-parameters and network architecture specified previously.

Impact of different proxy shapes. We first experiment with two simple proxy shapes: a cuboid or an ellipsoid. As shown in Table 2, using the cuboid proxy shape yields superior performance on all datasets. To get more insights, we also try to predict 3D coordinates aligned with the object's surface, *i.e*. we try to predict rendered coordinates of a 3D mesh model given cuboid proxy shapes as input reference poses. In this case, we exclude the OnePose dataset from the training set, since no mesh model is available. Interestingly, this cuboid-to-mesh setting performs much worse than cuboid-to-cuboid, meaning that it is easier for the network to regress 3D coordinates of an invisible cuboid (not necessarily aligned with the object surface) than actually reconstruct the object's unknown 3D shape. In other words, the model *does not need* to know nor infer the 3D object shape to estimate its pose. We use the cuboid-to-cuboid setting in all subsequent experiments.

Training data ablation. We then conduct an ablation to measure the importance of diversity in the training data. To that aim, we discard parts of the training set, still ensuring that all models trains for the same number of steps in each setting for the sake of fair comparison. Table 3 shows that having more diversity in the training set is critical to improve performance on all test sets. This result suggests that, despite the great diversity between datasets (for instance, ABO contains mostly furnitures), knowledge can effectively be shared and transferred between datasets.

Impact of the number of reference images. Increasing the number of reference views K at test time leads to better performances. We achieve accuracy scores of 68.4, 75.5 and 78.4% with $K = 16$, 32 and 64 respectively on LINEMOD. We observe similar behaviors on OnePose (with 87.8, 88.4 and 88.6% accuracy at 5cm-5deg resp.) and on ABO (74.8, 76.9 and 77.0% accuracy at 5cm-5deg resp.). This is expected because the model is more likely to find useful information in at least one reference view if we increase the number of these. It also shows that a model trained with a given number of reference views $K = 32$ at train time can generalize to a different number of reference views at test time.

Impact of pretraining. We finally assess the benefit of preemptively pretraining the network with a self-supervised objective. We specifically investigate whether pretraining is beneficial, and in particular, whether it should be *geometrically-oriented* or not. We thus compare CroCo pretraining (Weinzaepfel et al. 2022b) with MAE pretraining (He et al. 2021). The latter yields state-of-the-art results in many vision tasks, and is in addition compatible with our ViT-based architecture.

Name	Object Name													Avg.
	ape	benchwise	cam	can	cat	driller	duck	eggbox*	glue*	holepuncher	iron	lamp	phone	
	ADD(S)-0.1d													
Gen6D	-	62.1	45.6	-	40.9	48.8	16.2	-	-	-	-	-	-	-
OnePose	11.8	92.6	88.1	77.2	47.9	74.5	34.2	71.3	37.5	54.9	89.2	87.6	60.6	63.6
OnePose++	31.2	97.3	88.0	89.8	70.4	92.5	42.3	99.7	48.0	69.7	97.4	97.8	76.0	76.9
Ours ($K = 16$)	39.4	64.6	73.1	76.3	63.0	83.5	43.4	99.2	61.3	83.7	72.1	84.1	45.1	68.4
Ours ($K = 64$)	47.2	73.5	87.5	85.4	80.2	92.4	60.8	99.6	69.7	93.5	82.4	95.8	51.6	**78.4**
	Proj2D													
OnePose	35.2	94.4	96.8	87.4	77.2	76.0	73.0	89.9	55.1	79.1	92.4	88.9	69.4	78.1
OnePose++	97.3	99.6	99.6	99.2	98.7	93.1	97.7	98.7	51.8	98.6	98.9	98.8	94.5	94.3
Ours ($K = 16$)	96.6	82.9	95.1	92.7	95.4	89.9	89.4	98.6	94.0	98.5	79.1	85.2	76.0	90.3
Ours ($K = 64$)	97.1	94.1	98.4	98.2	98.4	95.7	96.3	99.0	94.8	99.3	94.6	94.2	88.9	**96.1**

Table 1: Results on LINEMOD and comparison with other *one-shot* baselines. Symmetric objects are indicated by *.

Proxy shape (input→output)	LINEMOD ADD(s)-0.1d↑	OnePose 5cm-5deg ↑	ABO 5cm-5deg ↑
ellipsoid → ellipsoid	58.0	79.9	70.8
cuboid → cuboid	**60.9**	**88.3**	**74.4**
cuboid → mesh	42.3	40.4	63.7

Table 2: Impact of the 3D proxy shape.

Training Dataset	LINEMOD ADD(s)-0.1d↑	OnePose 5cm-5deg ↑	ABO 5cm-5deg ↑
BOP	44.2	72.0	3.4
OnePose	4.9	66.2	0.8
ABO	20.6	61.7	70.2
BOP + OnePose	49.5	83.2	5.62
BOP + OnePose + ABO	**60.9**	**88.3**	**74.4**

Table 3: Ablation on training datasets.

Pre-training	LINEMOD		OnePose	
	ADD(S)-0.1d ↑	Proj2D ↑	3cm-3deg ↑	5cm-5deg ↑
None	16.6	27.3	27.5	54.8
MAE	39.4	56.7	54.0	72.3
Croco	**68.4**	**90.3**	**76.3**	**87.8**

Table 4: Impact of the pre-training strategy.

Contrary to CroCo, however, MAE has no explicit relation to 3D geometry. We present results in Table 4. We first note a considerable drop in performance when the network is trained from scratch (*i.e*. no pretraining). We then observe that, while MAE pretraining does improve a lot over no pretraining at all, it is still largely behind the performance attained by CroCo pretraining. Note that there is no unfair advantage in using CroCo, since CroCo is *not* trained on any object-centric data. Rather, CroCo pretraining data includes scene-level and landmark-level indoor and outdoor scenes, such as Habitat, MegaDepth, etc. Note that we systematically measure generalization performance (*i.e*. testing on unseen objects), hence clearly demonstrating how geometry-oriented pretraining is crucial for generalization.

Visualization. To understand how the network works internally, we visualize interactions happening in the cross-attention of the decoder in Figure 4. Undeniably, the model does perform matching under the hood to solve the task, as we see that all interactions consist of token-level correspondences between their corresponding patches. This is interesting, because the network is never explicitly trained for establishing correspondences. This also explains why the CroCo pretraining is so important, as this latter essentially consists in learning to establish correspondences between different viewpoints, see (Weinzaepfel et al. 2022b).

Comparison with the State of the Art

LINEMOD. We compare against Gen6D (Liu et al. 2023), OnePose (Sun et al. 2022) and OnePose++ (He et al. 2023), which are one-shot methods similar to our approach on the *ADD(S)-0.1d* and *Proj2D* metrics. As shown in Table 1, our approach outperforms these one-shot methods. Compared to the other one-shot methods, it is noteworthy that our method does not require any knowledge of the 3D object shape as input, in contrast to OnePose and OnePose++ which reconstruct 3D SfM model in advance. Our method gives 1.5% and 1.8% improvements on the ADD-S and Proj2D metrics, respectively, compared to the best competitor.

OnePose and OnePose-LowTexture. We again compare our approach with OnePose and OnePose++ (Sun et al. 2022; He et al. 2023), as well as some SfM baselines, on the challenging OnePose test set, which has the particularity of not providing mesh models. Results are provided in Table 6 in terms of the standard *cm-degree* accuracy for different thresholds. Note that "HLoc (LoFTR*)" uses LoFTR coarse matches for SfM and uses full LoFTR to match the query image and its retrieved images for pose estimation. Our method lags behind OnePose++ at the tightest 1cm/1deg threshold. In contrast to methods based on establishing pixel correspondences, such as OnePose++, which can be pixel-precise, and therefore provide high-precision pose estimates, our method predicts the coordinates of an 'invisible' proxy shape. This is definitely harder, and as a result, the resulting pose estimate is noisier. However, as the accuracy threshold of the

	K	LINEMOD		OnePose dataset			OnePose-LowTexture		
		ADD(s)-0.1d↑	Proj2D ↑	1cm-1deg	3cm-3deg	5cm-5deg	1cm-1deg	3cm-3deg	5cm-5deg
OnePose++	8	10.3	10.4	**36.1**	62.4	67.9	4.2	13.9	18.5
Ours		**55.5**	**75.9**	25.0	**72.6**	**85.7**	**9.7**	**44.8**	**65.2**
OnePose++	16	35.2	57.9	**46.6**	76.1	82.8	12.1	39.2	51.6
Ours		**68.4**	**90.3**	28.5	**76.3**	**87.8**	**12.4**	**51.3**	**71.9**
OnePose++	32	56.7	82.1	**49.7**	**78.6**	85.4	**16.8**	52.9	67.0
Ours		**75.5**	**94.7**	29.6	77.6	**88.4**	14.1	**53.6**	**73.4**
OnePose++	64	56.8	90.2	**50.6**	**80.0**	86.6	**16.8**	**56.2**	71.1
Ours		**78.4**	**96.1**	30.0	78.0	**88.6**	14.1	54.3	**74.2**
OnePose++	All	76.9	94.3	51.1	80.8	87.7	16.8	57.7	72.1

Table 5: Comparison of our model and OnePose++ with restricted numbers of reference images K.

	SfM	OnePose dataset			OnePose-LowTexture		
		1cm-1deg	3cm-3deg	5cm-5deg	1cm-1deg	3cm-3deg	5cm-5deg
HLoc *(SPP + SPG)*	yes	51.1	75.9	82.0	13.8	36.1	42.2
HLoc *(LoFTR*)*	yes	39.2	72.3	80.4	13.2	41.3	52.3
OnePose	yes	49.7	77.5	84.1	12.4	35.7	45.4
OnePose++	yes	**51.1**	**80.8**	87.7	**16.8**	**57.7**	72.1
Ours ($K = 16$)	no	28.5	76.3	87.8	12.4	51.3	71.9
Ours ($K = 64$)	no	30.0	78.0	**88.6**	14.1	54.3	**74.2**

Table 6: Comparison with *One-shot* Baselines. Our method is compared with HLoc (Sarlin et al. 2018) combined with different feature matching methods (Sarlin et al. 2019; Sun et al. 2021), OnePose (Sun et al. 2022) and OnePose++ (He et al. 2023). We denote as 'SfM' methods relying on an explicit 3D reconstruction of the objects.

performance metric increases (5cm/5deg), our method outperforms correspondence-based methods, demonstrating better robustness overall to challenging conditions.

Limited number of reference images. We also compare with OnePose++ in scenarios where the number of available reference images is limited. We experiment with various settings by altering the number of reference images (K) and report results in Table 5. In the case of OnePose++, the 'All' configuration entails using 170 and 130 reference images on average on the LINEMOD and OnePose datasets, respectively. It is noteworthy that as K decreases to values below 32, the performance of OnePose++ significantly drops on both LINEMOD and OnePose-LowTexture datasets. In contrast, our method exhibits a steady performance with only marginal degradation in accuracy. This result demonstrates the superior robustness of our approach in situations where the number of available reference images is limited.

We point out that our method is more practical than OnePose++, since it indiscriminately takes videos or small image sets with camera poses as raw inputs. In comparison, OnePose++ relies on videos and SfM pre-processing to build 3D object representations (we note they also rely on ground-truth poses from ARKit-scene), which is slow, complex and prone to failure – all of this strongly impairing scalability.

Detailed Timings

We report median computation times obtained with a NVIDIA V100 GPU, repeating measures 10 times for robustness. At inference, our model takes 20.66ms to process a single query image assuming $K = 16$ pre-encoded reference views (62.64ms for $K = 32$ respectively). This is 3 4 times faster than OnePose (Sun et al. 2022) and OnePose++ (He et al. 2023) whose 2D-3D matching modules take 66.4 and 88.2ms respectively.

Conclusion

We propose a novel approach, called MFOS, for model-free one-shot object pose estimation. In contrast to existing one-shot methods, MFOS does not need any 3D model of the target object, such as a mesh or point-cloud, and only requires a set of reference images annotated with the object poses and its approximate dimensions. It is able to implicitly extract 3D information from reference images, jointly matching, combining and extrapolating pose information with the query image, using only generic modules from a ViT architecture. In contrast to all existing methods, our approach is inherently simple, practical and scalable. In an extensive ablative study, we have determined good practices with this novel type of architecture in the field. Experiments show that our approach outperforms existing one-shot methods and show significant robustness in scenarios with a limited number of reference images.

Acknowledgments

This work was supported in part by IITP/NRF grants funded by the Korean government (MSIT, 2021-0-00105

Development of Model Compression Framework for Scalable On-Device AI Computing on Edge Applications, 2021-0-02068 Artificial Intelligence Innovation Hub, NRF-2021M3F3A2A02037893), and Inter-university Semiconductor Research Center (ISRC), SNU.

References

Belghit, H.; Bellarbi, A.; Zenati, N.; and Otmane, S. 2018. Vision-based Pose Estimation for Augmented Reality : A Comparison Study. *CoRR*, abs/1806.09316.

Brachmann, E.; Krull, A.; Michel, F.; Gumhold, S.; Shotton, J.; and Rother, C. 2014. Learning 6D Object Pose Estimation Using 3D Object Coordinates. In Fleet, D.; Pajdla, T.; Schiele, B.; and Tuytelaars, T., eds., *Computer Vision – ECCV 2014*, 536–551. Cham: Springer International Publishing. ISBN 978-3-319-10605-2.

Brachmann, E.; Michel, F.; Krull, A.; Yang, M. Y.; Gumhold, S.; and Rother, C. 2016. Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3364–3372.

Brégier, R. 2021. Deep regression on manifolds: a 3D rotation case study. *CoRR*.

Cai, D.; Heikkilä, J.; and Rahtu, E. 2022. OVE6D: Object Viewpoint Encoding for Depth-based 6D Object Pose Estimation. arXiv:2203.01072.

Chen, D.; Li, J.; and Xu, K. 2020. Learning Canonical Shape Space for Category-Level 6D Object Pose and Size Estimation. *CoRR*, abs/2001.09322.

Chen, H.; Wang, P.; Wang, F.; Tian, W.; Xiong, L.; and Li, H. 2022. EPro-PnP: Generalized End-to-End Probabilistic Perspective-n-Points for Monocular Object Pose Estimation. arXiv:2203.13254.

Chen, K.; and Dou, Q. 2021. SGPA: Structure-Guided Prior Adaptation for Category-Level 6D Object Pose Estimation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2753–2762.

Chen, W.; Jia, X.; Chang, H. J.; Duan, J.; Shen, L.; and Leonardis, A. 2021. FS-Net: Fast Shape-based Network for Category-Level 6D Object Pose Estimation with Decoupled Rotation Mechanism. *CoRR*, abs/2103.07054.

Chen, X.; Dong, Z.; Song, J.; Geiger, A.; and Hilliges, O. 2020. Category Level Object Pose Estimation via Neural Analysis-by-Synthesis. *CoRR*, abs/2008.08145.

Collins, J.; Goel, S.; Deng, K.; Luthra, A.; Xu, L.; Gundogdu, E.; Zhang, X.; Yago Vicente, T. F.; Dideriksen, T.; Arora, H.; Guillaumin, M.; and Malik, J. 2022. ABO: Dataset and Benchmarks for Real-World 3D Object Understanding. *CVPR*.

Deng, X.; Xiang, Y.; Mousavian, A.; Eppner, C.; Bretl, T.; and Fox, D. 2019. Self-supervised 6D Object Pose Estimation for Robot Manipulation. *CoRR*, abs/1909.10159.

Do, T.-T.; Pham, T. T.; Cai, M.; and Reid, I. D. 2018. LieNet: Real-time Monocular Object Instance 6D Pose Estimation. In *British Machine Vision Conference*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*.

Eldar, Y.; Lindenbaum, M.; Porat, M.; and Zeevi, Y. 1997. The farthest point strategy for progressive image sampling. *IEEE Transactions on Image Processing*.

Gou, M.; Pan, H.; Fang, H.-S.; Liu, Z.; Lu, C.; and Tan, P. 2022. Unseen Object 6D Pose Estimation: A Benchmark and Baselines. arXiv:2206.11808.

He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; and Girshick, R. B. 2021. Masked Autoencoders Are Scalable Vision Learners. *CoRR*, abs/2111.06377.

He, X.; Sun, J.; Wang, Y.; Huang, D.; Bao, H.; and Zhou, X. 2023. OnePose++: Keypoint-Free One-Shot Object Pose Estimation without CAD Models. arXiv:2301.07673.

He, Y.; Wang, Y.; Fan, H.; Sun, J.; and Chen, Q. 2022. FS6D: Few-Shot 6D Pose Estimation of Novel Objects. arXiv:2203.14628.

Henriques, J.; Martins, P.; Caseiro, R.; and Batista, J. 2014. Fast training of pose detectors in the fourier domain. *Advances in Neural Information Processing Systems*, 4: 3050–3058.

Hietanen, A.; Latokartano, J.; Foi, A.; Pieters, R.; Kyrki, V.; Lanz, M.; and Kämäräinen, J. 2019. Benchmarking 6D Object Pose Estimation for Robotics. *CoRR*, abs/1906.02783.

Hinterstoisser, S.; Cagniart, C.; Ilic, S.; Sturm, P.; Navab, N.; Fua, P.; and Lepetit, V. 2012a. Gradient Response Maps for Real-Time Detection of Textureless Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(5): 876–888.

Hinterstoisser, S.; Holzer, S.; Cagniart, C.; Ilic, S.; Konolige, K.; Navab, N.; and Lepetit, V. 2011. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *2011 International Conference on Computer Vision*, 858–865.

Hinterstoisser, S.; Lepetit, V.; Ilic, S.; Holzer, S.; Bradski, G. R.; Konolige, K.; and Navab, N. 2012b. Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes. In *Asian Conference on Computer Vision*.

Hodan, T.; Michel, F.; Brachmann, E.; Kehl, W.; Buch, A. G.; Kraft, D.; Drost, B.; Vidal, J.; Ihrke, S.; Zabulis, X.; Sahin, C.; Manhardt, F.; Tombari, F.; Kim, T.; Matas, J.; and Rother, C. 2018. BOP: Benchmark for 6D Object Pose Estimation. *CoRR*, abs/1808.08319.

Iwase, S.; Liu, X.; Khirodkar, R.; Yokota, R.; and Kitani, K. M. 2021. RePOSE: Real-Time Iterative Rendering and Refinement for 6D Object Pose Estimation. *CoRR*, abs/2104.00633.

Jocher, G.; Stoken, A.; Borovec, J.; NanoCode012; ChristopherSTAN; Changyu, L.; Laughing; tkianai; Hogan, A.; lorenzomammana; yxNONG; AlexWang1900; Diaconu, L.; Marc; wanghaoyang0106; ml5ah; Doug; Ingham, F.; Frederik; Guilhen; Hatovix; Poznanski, J.; Fang, J.; , L. Y.; changyu98; Wang, M.; Gupta, N.; Akhtar, O.; PetrDvoracek; and Rai, P. 2020. ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements.

Kehl, W.; Manhardt, F.; Tombari, F.; Ilic, S.; and Navab, N. 2017. SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again. *CoRR*, abs/1711.10006.

Kehl, W.; Tombari, F.; Navab, N.; Ilic, S.; and Lepetit, V. 2016. Hashmod: A Hashing Method for Scalable 3D Object Detection. arXiv:1607.06062.

Kendall, A.; Gal, Y.; and Cipolla, R. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*.

Lee, T.; Lee, B.; Kim, M.; and Kweon, I. S. 2021. Category-Level Metric Scale Object Shape and Pose Estimation. *CoRR*, abs/2109.00326.

Li, Z.; and Ji, X. 2020. Pose-guided Auto-Encoder and Feature-Based Refinement for 6-DoF Object Pose Regression. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 8397–8403.

Li, Z.; Wang, G.; and Ji, X. 2019. CDPN: Coordinates-Based Disentangled Pose Network for Real-Time RGB-Based 6-DoF Object Pose Estimation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 7677–7686.

Lin, Y.; Florence, P.; Barron, J. T.; Garcia, A. R.; Isola, P.; and Lin, T. 2020. iNeRF: Inverting Neural Radiance Fields for Pose Estimation. *CoRR*, abs/2012.05877.

Liu, Y.; Wen, Y.; Peng, S.; Lin, C.; Long, X.; Komura, T.; and Wang, W. 2023. Gen6D: Generalizable Model-Free 6-DoF Object Pose Estimation from RGB Images. arXiv:2204.10776.

Marchand, E.; Uchiyama, H.; and Spindler, F. 2016. Pose Estimation for Augmented Reality: A Hands-On Survey. *IEEE Transactions on Visualization and Computer Graphics*, 22.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. *CoRR*, abs/2003.08934.

Olson, E. 2011. AprilTag: A robust and flexible visual fiducial system. In *2011 IEEE International Conference on Robotics and Automation*, 3400–3407.

Park, K.; Mousavian, A.; Xiang, Y.; and Fox, D. 2019. LatentFusion: End-to-End Differentiable Reconstruction and Rendering for Unseen Object Pose Estimation. *CoRR*, abs/1912.00416.

Park, K.; Patten, T.; and Vincze, M. 2019. Pix2Pose: Pixel-Wise Coordinate Regression of Objects for 6D Pose Estimation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE.

Pavllo, D.; Tan, D. J.; Rakotosaona, M.-J.; and Tombari, F. 2023. Shape, Pose, and Appearance from a Single Image via Bootstrapped Radiance Field Inversion. arXiv:2211.11674.

Peng, S.; Liu, Y.; Huang, Q.; Bao, H.; and Zhou, X. 2018. PVNet: Pixel-wise Voting Network for 6DoF Pose Estimation. *CoRR*, abs/1812.11788.

Rad, M.; and Lepetit, V. 2017. BB8: A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects without Using Depth. *CoRR*, abs/1703.10896.

Rios-Cabrera, R.; and Tuytelaars, T. 2013. Discriminatively Trained Templates for 3D Object Detection: A Real Time Scalable Approach. *2013 IEEE International Conference on Computer Vision*, 2048–2055.

Sarlin, P.; Cadena, C.; Siegwart, R.; and Dymczyk, M. 2018. From Coarse to Fine: Robust Hierarchical Localization at Large Scale. *CoRR*, abs/1812.03506.

Sarlin, P.; DeTone, D.; Malisiewicz, T.; and Rabinovich, A. 2019. SuperGlue: Learning Feature Matching with Graph Neural Networks. *CoRR*, abs/1911.11763.

Schönberger, J.; and Frahm, J.-M. 2016. Structure-from-motion Revisited. In *CVPR*.

Shugurov, I.; Li, F.; Busam, B.; and Ilic, S. 2022. OSOP: A Multi-Stage One Shot Object Pose Estimation Framework. arXiv:2203.15533.

Song, J. 2017. Sliding window filter based unknown object pose estimation. In *2017 IEEE International Conference on Image Processing (ICIP)*, 2642–2646.

Su, J.; Lu, Y.; Pan, S.; Murtadha, A.; Wen, B.; and Liu, Y. 2021. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*.

Sun, J.; Shen, Z.; Wang, Y.; Bao, H.; and Zhou, X. 2021. LoFTR: Detector-Free Local Feature Matching with Transformers. *CoRR*, abs/2104.00680.

Sun, J.; Wang, Z.; Zhang, S.; He, X.; Zhao, H.; Zhang, G.; and Zhou, X. 2022. OnePose: One-Shot Object Pose Estimation without CAD Models. arXiv:2205.12257.

Tejani, A.; Tang, D.; Kouskouridas, R.; and Kim, T.-K. 2014. Latent-Class Hough Forests for 3D Object Detection and Pose Estimation. In Fleet, D.; Pajdla, T.; Schiele, B.; and Tuytelaars, T., eds., *Computer Vision – ECCV 2014*, 462–477. Cham: Springer International Publishing. ISBN 978-3-319-10599-4.

Terzakis, G.; and Lourakis, M. 2020. A consistently fast and globally optimal solution to the perspective-n-point problem. In *ECCV*.

Tian, M.; Jr., M. H. A.; and Lee, G. H. 2020. Shape Prior Deformation for Categorical 6D Object Pose and Size Estimation. *CoRR*, abs/2007.08454.

Wang, G.; Manhardt, F.; Tombari, F.; and Ji, X. 2021. GDR-Net: Geometry-Guided Direct Regression Network for Monocular 6D Object Pose Estimation. *CoRR*, abs/2102.12145.

Wang, H.; Sridhar, S.; Huang, J.; Valentin, J.; Song, S.; and Guibas, L. J. 2019. Normalized Object Coordinate Space for Category-Level 6D Object Pose and Size Estimation. *CoRR*, abs/1901.02970.

Wang, J.; Chen, K.; and Dou, Q. 2021. Category-Level 6D Object Pose Estimation via Cascaded Relation and Recurrent Reconstruction Networks. *CoRR*, abs/2108.08755.

Weinzaepfel, P.; Arora, V.; Cabon, Y.; Lucas, T.; Brégier, R.; Leroy, V.; Csurka, G.; Antsfeld, L.; Chidlovskii, B.; and Revaud, J. 2022a. Improved Cross-view Completion Pre-training for Stereo Matching. *arXiv preprint arXiv:2211.10408*.

Weinzaepfel, P.; Leroy, V.; Lucas, T.; Brégier, R.; Cabon, Y.; Arora, V.; Antsfeld, L.; Chidlovskii, B.; Csurka, G.; and Revaud, J. 2022b. CroCo: Self-Supervised Pre-training for 3D Vision Tasks by Cross-View Completion. In *NeurIPS*.

Wen, B.; and Bekris, K. E. 2021. BundleTrack: 6D Pose Tracking for Novel Objects without Instance or Category-Level 3D Models. *CoRR*, abs/2108.00516.

Xiang, Y.; Schmidt, T.; Narayanan, V.; and Fox, D. 2017. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. *CoRR*, abs/1711.00199.

Xie, T.; Dai, K.; Wang, K.; Li, R.; and Zhao, L. 2023. DeepMatcher: A Deep Transformer-based Network for Robust and Accurate Local Feature Matching. *arXiv preprint arXiv:2301.02993*.

Zakharov, S.; Shugurov, I.; and Ilic, S. 2019. DPOD: Dense 6D Pose Object Detector in RGB images. *CoRR*, abs/1902.11020.