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

OSOP: A Multi-Stage One Shot Object Pose Estimation Framework

Ivan Shugurov[1,3], Fu Li[1,2], Benjamin Busam[1], Slobodan Ilic[1,3]

[1] TU Munich [2] NUDT [3] Siemens AG
{ivan.shugurov, fu.li, b.busam, slobodan.ilic}@tum.de

Abstract

We present a novel one-shot method for object detection and 6 DoF pose estimation, that does not require training on target objects. At test time, it takes as input a target image and a textured 3D query model. The core idea is to represent a 3D model with a number of 2D templates rendered from different viewpoints. This enables CNN-based direct dense feature extraction and matching. The object is first localized in 2D, then its approximate viewpoint is estimated, followed by dense 2D-3D correspondence prediction. The final pose is computed with PnP. We evaluate the method on LineMOD, Occlusion, Homebrewed, YCB-V and TLESS datasets and report very competitive performance in comparison to the state-of-the-art methods trained on synthetic data, even though our method is not trained on the object models used for testing.

1. Introduction

The rapid development of high quality 6 DoF pose estimation methods is underway. According to the BOP challenge [14], which combines publicly available 6 DoF pose estimation datasets and offers standardized evaluation and comparison procedures, the field is dominated by deep learning methods [2, 12, 16, 19, 21, 22, 22, 24–27, 36, 37, 47, 49–52, 59]. The methods' performance is, however, limited by the availability of labeled training data. Accurate 6 DoF pose annotation of real data is a complicated and time-consuming process [20], that must be repeated manually for each new object. This severely limits the practical applicability of 6 DoF pose estimation methods. Additionally, labels can exhibit imperfection [4]. As synthetic data preparation tools improved, more methods shifted to training on synthetically rendered images. This greatly simplifies the preparation of data for new objects. These time-consuming and computationally-intensive data rendering and model training steps, on the other hand, must be repeated for each new target object of interest.

While one-shot object detection, i.e., detection of novel objects not seen during training, appears to yield promising results for conventional 2D detection, its extensions to pose estimation were very limited. There are very few related works that attempt to generalize to new objects. They primarily focus on objects from the same category [6, 31, 55], objects with very similar geometry [38, 39], rely on partially training on target objects [52], or limit the task to viewpoint estimation [57]. We extend the one-shot object detection ideas to estimate the full 6 DoF pose. Our method is trained only once and then automatically generalizes to new objects without training on them, obviating the need for synthetic or real data preparation and training for new objects.

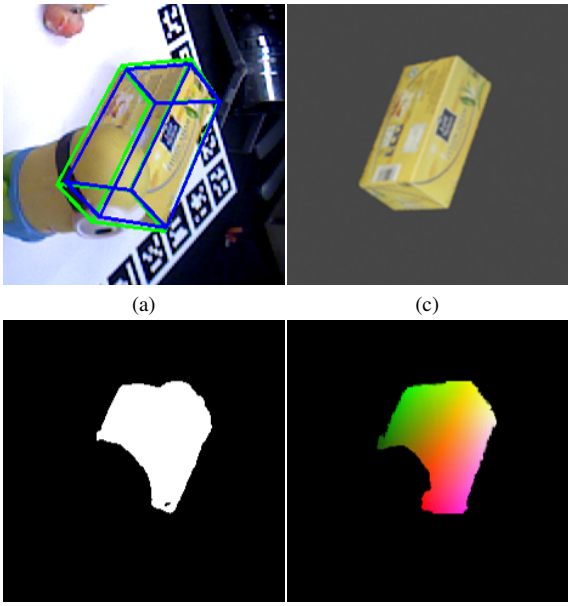


Figure 1. Qualitative evaluation of the proposed method on an object from the Homebrewed dataset [20]. a) an input image, cropped only for visualization purposes, with a comparison of the ground truth (green cuboid) and estimated (blue cuboid) poses; b) predicted one-shot segmentation ; c) a matched template; and d) predicted correspondences as color-coded NOCS correspondences.

The 4-stage pipeline of the proposed approach is visualized in Figure 2. The input to the method is a test image

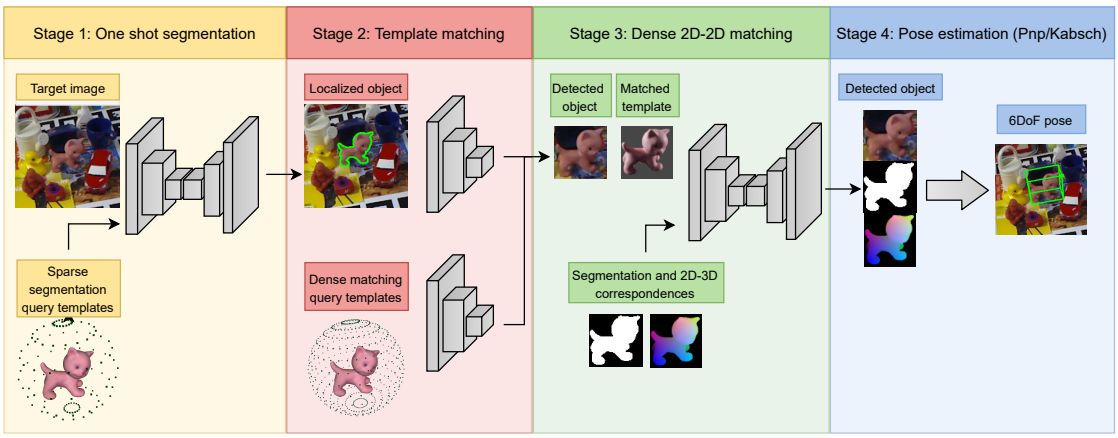


Figure 2. **Pipeline of the proposed detector.** 1) One-shot object localization conditioned on the 3D model. 2) Initial viewpoint estimation by template matching. 3) Dense 2D-2D matching between the image patch and the matched template. 4) 6 DoF pose estimation with PnP+RANSAC or Kabsch+RANSAC. The proposed pose estimation pipeline generalizes well to new target objects not seen duing training.

and a textured 3D model of the target object of interest. Inspired by OS2D [35], the method relies on dense sliding window-based feature correlation between the object, represented with a sparse set of 2D templates obtained by rendering the object from various viewpoints, and the input test image. In the first stage, one-shot object segmentation is performed. The detected object is matched to a database of object renderings in the second stage to perform initial viewpoint estimation. In the third stage, a network estimates dense 2D-2D correspondences between pixels of the input image patch and the matched template, whose pose is known. This provides us with 2D-3D correspondences between the pixels of the input image and the 3D model. This enables 6 DoF pose estimation using PnP [23] or Kabsch [1] with RANSAC [8] in the last stage.

The evaluation of our approach on five datasets (LineMOD, Occlusion, HomebrewDB, YCB-V and TLESS) proves that it fully generalizes to new objects and scenes not seen during training. Our key contributions include: 1) The first RGB-based one-shot pose estimation pipeline that truly scales to new objects without training on them. This results in a considerable time reduction required for synthetic data generation and retraining. 2) A novel architecture and a novel attention mechanism for one-shot semantic segmentation; 3) An architecture for dense 2D-2D matching that enables 2D-3D correspondence transfer from a template with known 6 DoF pose to a target image with unknown pose.

2. Related Work

State of the Art DL Methods. The main trend is to predict the locations of 2D keypoints for which their corresponding points on the 3D model are known. In particular, IPose [16], YOLO6D [53], PVNet [37], HybridPose [50] and [19] predict a sparse set of the pre-defined keypoints. DPOD [48, 49, 59], CDPN [27], Pix2Pose [36], and EPOS [12] rely on dense pixel-wise 2D-3D correspondence estimation. Pose is then estimated using the PnP [23] algorithm. Prediction of keypoints allows for more robust pose estimation and explicitly uses handling of occlusions. A notable exception is CosyPose [22], where excellent pose estimation results are obtained with direct pose regression followed by a multi-view refinement.

Generalization to Novel Objects. Point Pair Features (PPF) [7] is arguably the only commercially available [34] application-ready one-shot approach for object detection and pose estimation. It works by approximating local geometry with oriented point pairs and exhaustive feature matching between the scene and the object model. PPF-based methods led the BOP challenge [14] until very recently, when they were outperformed by deep learning methods. The disadvantages of the PPF methods is their dependence on depth information with per-point normals and a slow run time, which limits the potential applications.

Several attempts have been made to make deep learning-based pose estimation methods generalizable to new objects. NOCS [55], followed by [6, 24, 31], proposed training a network to predict the poses of objects in a specific narrow class. Despite advancements, the performance is still entirely dependent on the degree of similarity of objects within the object category. On the other hand, we make no such assumption because our approach explicitly employs the 3D object model during inference. Another line of research attempts to capitalize on similarities between objects in training and testing sequences. CorNet [39] used only corners to approximate object geometry. Dense local shape descriptors for each object pixel were predicted in [38], which could be matched with the object model. These methods needed a high degree of similarity between training and testing objects to work well. Sundermeyer et al. [51] demonstrate that rotation

estimation by template matching can generalize to new objects that were not seen during training, even if the feature extractor was trained on objects from a different dataset. The method, however, trains a 2D object detector on the target object classes. In contrast to them, our proposed method is fully one-shot and does not require any training on the target objects. The method of [57] is more similar to our method. Here, the viewpoint is directly predicted by concatenating features extracted from the 3D model and the image. However, the method assumes that the object is already perfectly localized in 2D and does not estimate the full 6 DoF pose.

One-Shot Methods. 2D object detectors [9, 10, 28, 29, 40, 42, 54] show excellent results when trained on target objects but cannot detect novel objects by design. One-shot methods [5, 15, 17, 18, 32] focus on generalization and allow for object detection of a new object, represented by a single template, in an input image. A common design principle is to extract features from a template and a target image using a Siamese network and then match them to localize the object. The most related paper to our approach is OS2D [35], which is built around the ideas of feature matching for image alignment [43–45]. It proposed to correlate template and image features using the sliding window approach. The key difference between our method and one-shot object detection methods is that we aim to explicitly use the full 3D information of the object mode.

3. Methodology

One-shot methods for 2D object detection use a target RGB image and a query template of the object of interest as input during inference, neither of which was seen during training. The inputs in our method are a target RGB image and a 3D model of the object. The 3D model is represented by a set of 2D query templates rendered from various virtual camera viewpoints placed on a sphere around the object. Our pipeline consists of four stages as summarized in Figure 2 with stages object segmentation (1), template matching (2), 2D-2D matching (3), and pose estimation (4). Stages 3 and 4 can potentially be executed several times to produce multiple pose hypotheses.

In the following, we use I of size $H \times W$ to denote an image, which implicitly depends on the object model $\mathcal{M}$ and its pose $T \in SE(3)$. A feature extractor $F^k_{FE}(I) \in \mathbb{R}^{H^k \times W^k \times D^k}$ uses a pre-trained network to extract feature maps of depth dimension D^k from the input image. In the paper, we use pre-computed feature maps from several depth levels of the network, which are indexed by $\{k \in \mathbb{N} \mid 1 \leq k \leq N\}$. $\bar{F}^k_{FE}(I) \in \mathbb{R}^{D^k}$ stands for a feature extractor which extends F^k_{FE} by spatial averaging along height and width for each depth dimension of the feature map to produce a single vector of length D^k.

3.1. One-Shot Segmentation

The first stage network takes an image and a descriptor of the 3D model and predicts a binary segmentation mask indicating the location of the object' visible part. The core idea is to describe a textured CAD model using a set of viewpoint-based templates generated by rendering the object in various rotations. This brings the problem closer to the standard 2D one-shot methods. The key difference is that we compute a single descriptor based on pre-rendered templates, allowing the image to be matched to all of the templates in a single shot, as opposed to the standard one-shot object detection, which treats each query template independently. Figure 3 depicts the overall architecture of the network.

We build on the concept of dense feature matching first, which was first proposed in [43] and later extended for the task of object detection in [35]. The basic idea is to compute per-pixel correlations between feature maps of the target image and the features from the object descriptor. Feature precomputation is visualized in Figure 4. Pre-computed image features $\mathbf{f}^k = F^k_{FE}(I) \in \mathbb{R}^{H^k \times W^k \times D^k}$ and a 4D descriptor tensor $\mathbf{o}^k = F_{FE}(\mathcal{M}) \in \mathbb{R}^{X^k \times Y^k \times Z^k \times D^k}$ for the object $\mathcal{M}$ are compared. The 4D descriptor tensor $\mathbf{o}^k$ collects all templates rendered from the virtual viewpoints on the sphere around the object, where the first two dimensions (X, Y) stand for a camera position w.r.t. the object coordinate system using polar coordinates, while the third dimension Z stands for in-plane rotations. The 4th dimension D^k refers to feature maps extracted from each template at multiple depth levels of the neural network indexed by k. Each element of the tensor is one viewpoint template represented with the corresponding feature vector. It is defined as $\mathbf{o}^k_{x,y,z} = \bar{F}^k_{FE}(I(R(x,y,z) \cdot \mathcal{M}))$, where R is a rotation matrix representing a virtual viewpoint on the sphere. Each pixel in the feature map $\mathbf{f}^k$ is matched to the entire object descriptor $\mathbf{o}^k$, resulting in a correlation tensor $\mathbf{c}^k \in \mathbb{R}^{H^k \times W^k \times X^k \times Y^k \times Z^k}$. For a particular pixel (h, w), the correlation tensor value is defined as

$$\mathbf{c}^k_{h,w,x,y,z} = \text{corr}\left(\mathbf{f}^k_{h,w}, \mathbf{o}^k_{x,y,z}\right), \tag{1}$$

where corr denotes Pearson correlation. The correlation tensor is then flattened to a 3D tensor $\mathbf{c}^k \in \mathbb{R}^{H^k \times W^k \times \left(X^k Y^k Z^k\right)}$. This way, each pixel of the target image feature map gets the list of all correlations of its feature vector with all the feature vectors of the descriptor.

The flattened correlation tensor is used in two ways. First, pre-computed correlations are used directly as the input to the decoder, as in [35, 43–45]. For that, the tensor $\mathbf{c}^k$ is processed by a 1×1 convolutional layer to reduce the number of dimensions from $\left(X^k Y^k Z^k\right)$ to L^k. The

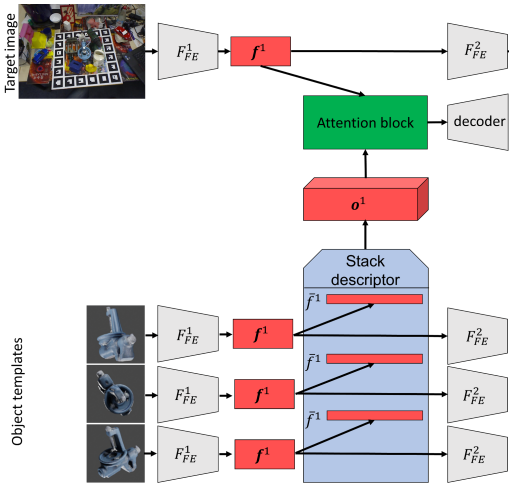


Figure 3. Encoder of the stage 1 network. The network takes an input image and an object model, represented by a sparse set of templates, and outputs a binary segmentation of the target image. The full detailed architecture is provided in the supplementary materials.

correlations are also used to compute pixel-wise attention. Pixel-wise attention allows us to effectively incorporate the original image features into the feature tensor and use them for more precise segmentation.

Raw pixel-wise attention at the feature map level k is defined simply as a sum of all $\left(X^k Y^k Z^k\right)$ correlations for a given pixel as

$$A^k_{h,w} = \max\left\{0, \sum_{j=1}^{(X^k Y^k Z^k)} \mathbf{c}^k_{h,w,j}\right\}. \tag{2}$$

Since simple attention can be very noisy in the early layers compared to later layers, we propose to condition per-pixel attention of each particular level k on the attention from the last level k_l, which tend to be more precise but have low resolution, as $\hat{A}^k_{h,w} = A^k_{h,w} \bigtriangledown \left(A^{k_l}\right)_{h',w'}$. $\bigtriangledown$ denotes a bilinear upsampling of the attention A^{k_l} to the size of A^k. The values are then filtered by zeroing out attention values below the average value, resulting in cleaner and more precise attention maps, as illustrated in the supplementary material:

$$\hat{A}^k_{h,w} = \begin{cases} \hat{A}^k_{h,w}, & \text{if } \hat{A}^k_{h,w} > \text{avg}_{h',w'} \hat{A}^k_{h',w'} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

A^{k_l} itself is thresholded but not conditioned on anything. All of the values are scaled to fall between 0 and 1. Image features are transformed using the attention maps as follows:

$$\hat{\mathbf{f}}^k_{h,w} = \hat{A}^k_{h,w} \cdot \mathbf{f}^k_{h,w} - (1 - \hat{A}^k_{h,w}) \cdot \mathbf{f}^k_{h,w}. \tag{4}$$

The attended features are then processed by a 1×1 convolutional layer to reduce dimensionality. Stacked $\hat{\mathbf{f}}^k$ and $\mathbf{c}^k$ are used jointly by the subsequent layers. Overall, the decoder resembles the UNet [46] approach of feature maps upsampling followed by convolutional layers until the initial image size is reached. The main distinction is that the network employs stacked $\hat{\mathbf{f}}^k$ and $\mathbf{c}^k$ at each level rather than skip connections. The network is trained to predict per-pixel probability that a pixel contains a visible part of the object. We used the Dice loss $\mathcal{L}_{Dice}$ [33] to handle imbalanced class data.

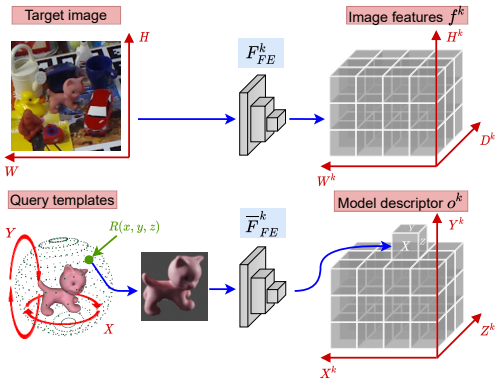


Figure 4. Feature computation for a target image and a 3D model. The top row illustrates how an input target RGB image is converted to a 3D feature tensor f^k. The bottom row demonstrates how object templates sampled along azimuth (axis X), elevation (Y) and in-plane rotation axis Y are transformed to a corresponding dense 4D model descriptor o^k.

3.2. Template Matching

For the initial viewpoint estimation, we rely on template matching via deep manifold learning , which has been shown to scale well to a large number of objects [3, 58] and to generalize to new objects [51] not seen during training. We rely on the same feature extraction network F_{FE} but use only the features from the last layer. We also add one 1×1 convolutional layer to decrease the dimensions from $H \times W \times D$ to $H \times W \times D'$. Template features $\mathbf{t} \in \mathbb{R}^{H \times W \times D'}$ and image features $\mathbf{f} \in \mathbb{R}^{H \times W \times D'}$ are pre-computed from the foregrounds of the query templates and from the foreground of the detected object in the target image denoted with using the segmentation predicted in the previous step respectively. Analogously to the first stage, similarity of two patches is estimated by computing per-pixel correlations between $\mathbf{f}$ and $\mathbf{t}$ using

$$\text{sim}\left(\mathbf{f}, \mathbf{t}\right) = \sum_{h,w} \text{corr}\left(\mathbf{f}_{h,w}, \mathbf{t}_{h,w}\right). \tag{5}$$

We train the network to increase similarity for patches which depict objects with very close rotations and at the same time penalize similarity for distant rotations. A modified triplet loss with dynamic margin is leveraged by opti-

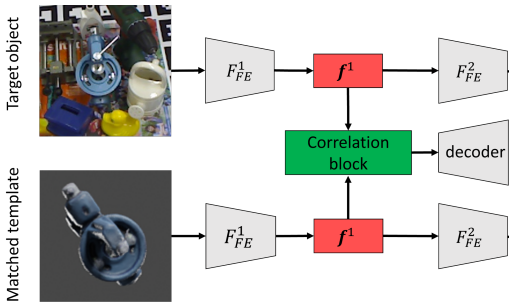


Figure 5. Encoder of the stage 3 network. The network takes an input image with the detected object and a matched template. Its output is a pixel-wise binary segmentation and dense 2D-2D correspondences. A detailed architecture is provided in the supplementary materials.

mizing

$$\mathcal{L}_{triplets} = \max\left\{0, 1 - \frac{\text{sim}(\mathbf{f}^{anchor}, \mathbf{f}^{+})}{\text{sim}(\mathbf{f}^{anchor}, \mathbf{f}^{-}) + m}\right\}, \quad (6)$$

where m is set to the angle between rotations of the object in the anchor and the puller patches. Using the terminology from [58], $\mathbf{f}^{anchor}$ is a descriptor of a randomly chosen object patch. $\mathbf{f}^{+}$ corresponds to a puller - a template in the pose very similar to the pose in the anchor, while $\mathbf{f}^{-}$ corresponds to the pusher with a dissimilar pose. A query template with the highest similarity to the detected object in the target image is chosen as a match at test time.

3.3. One-Shot Dense Correspondence Estimation

The goal of this stage is to establish 2D-3D correspondences between the image pixels and the object model. After the previous step, we have a patch with the detected object in unknown pose and a matched template in known pose. Establishing dense 2D-2D correspondences between the object patch and the template explicitly provides 2D-3D matches between the object pixels and the 3D object model. The correspondences can then be used to estimate the pose with PnP+RANSAC or Kabsch+RANSAC.

Similarly to the previous stages, the architecture of the 2D-2D matching follows the general idea of dense feature matching. Each pixel of the feature map $\mathbf{f}^k$, representing the input image patch of the detected object, is matched with all pixels of the template feature map $\mathbf{t}^k$ to form the correlation tensor $\mathbf{c}^k$. The network then predicts three values for each pixel: a binary foreground/background segmentation mask and a coordinate of the corresponding pixel on the template. Figure 5 depicts the architecture.

During training, a random object crop I_{obj} with its associated pose $T_{obj} \in SE(3)$ is sampled from a synthetic dataset. Then, a random template I_{tmp} is picked together with its pose $T_{tmp} \in SE(3)$, so that T_{obj} and T_{tmp} are relatively close. Availability of object poses allows us to compute per-pixel 2D-3D correspondence maps in both patches. Let $C : \mathcal{M} \times SE(3) \rightarrow [0,1]^{W \times H \times 3}$ denote 2D-3D correspondences for the object rendered in the given pose. Its inverse C^{-1} recomputes correspondences with respect to the unnormalized object coordinates, corresponding to the actual 3D object coordinates. It allows us to define a 2D correspondence pair distance in the model's 3D coordinate space:

$$d(p, p') \quad = \quad \left\| C^{-1}\left(I_{obj}\right)_p - C^{-1}\left(I_{tmp}\right)_{p'} \right\|_2 \quad (7)$$

where p and p' are pixel coordinates in the image and template patches respectively. Ground truth dense 2D-2D correspondences are established by matching pixel pairs corresponding to the closest points in the 3D coordinate system of the model. For a point $p \in \mathcal{I}_{obj}$ its corresponding template point is computed as $\underset{p' \in \mathcal{I}_{tmp}}{\operatorname{argmin}}\, d(p, p')$. We employ an outlier-aware rejection for 2D-2D correspondences with large 3D spatial discrepancy.

The segmentation loss is defined as a per-pixel Dice loss ($\mathcal{L}_{Dice}$). In addition, the network predicts a discrete 2D coordinate using a standard per-pixel cross-entropy classification loss denoted as $\mathcal{L}_{2D2D}$.

3.4. Pose Hypothesis Verification

We propose an optional step for generating and verifying pose hypotheses. Its goal is to reduce the imprecision caused by incorrect initial viewpoint estimation. Pose hypotheses are generated by independently estimating 2D-2D correspondences from each of the top N matched templates and estimating poses from them. We greedily remove matched templates that are too close to each other to reduce run times and ensure more diverse poses. In practice, we filtered matches using a 15-degree threshold and picked top 25 templates from them. If depth is available, hypotheses are ranked based on the quality of fit of observed and rendered depth. In the RGB case, per-pixel correspondence error between the predicted correspondences and the rendered object is measured.

4. Experiments

We evaluate the proposed method on Linemod [11] (LM), Occlusion [2] (LMO), Homebrewed [20] (HBD), YCB-V [56]. Evaluation on the TLESS dataset [13] is provided in the supplementary material. Each detector stage is trained separately for each target dataset, so that train and test objects differ. For example, we use train the networks used for experiments on the Linemod dataset using all objects from the Homebrewed and YCB-V datasets. Linemod's and Homebrewed's common objects are skipped during training. We used the synthetic PBR images provided by the organizers of the BOP challenge [14] in all experiments. We denote methods, that used only PBR [14]

Table 1. 2D detection results in comparison to YOLO [41], trained on target objects, and one shot OS2D [35] detectors on the BOP split of the test data [14]. Results on the Homebrewed dataset [20] are reported on the publicly available validation split.

Dataset	Method	Precision	Recall	F1	Best Recall
LM	YOLO	0.99	0.97	0.98	0.99
	Ours	0.47	0.86	0.61	0.86
	OS2D	0.28	0.2	0.23	0.57
LMO	YOLO	0.69	0.67	0.68	0.85
	Ours	0.31	0.61	0.41	0.61
	OS2D	0.16	0.31	0.21	0.53
HBD	YOLO	0.76	0.74	0.75	0.91
	Ours	0.43	0.73	0.54	0.73
	OS2D	0.24	0.29	0.26	0.44
YCB	YOLO	0.72	0.84	0.78	0.98
	Ours	0.41	0.8	0.54	0.8
	OS2D	0.12	0.18	0.14	0.26

synthetic images, with "PBR", methods with custom synthetic images as "synt" and methods, which used a mix of real and synthetic data, as "mix". The standard ADD score [7] with the 10% diameter threshold is reported for the Linemod dataset. The BOP Average Recall (AR) score [14] is reported for the other datasets.

4.1. 2D Object Localization

We compare our approach's 2D detection capabilities to two baselines. First, we compare it to OS2D [35], a state of the art one-shot object detection method. For fairness, we ran OS2D with exactly the same templates as our method. Second, we compare to YOLOv3 [41], which was trained separately for each scene using the synthetic PBR image [14]. It establishes the upper bound for the object detection performance and demonstrates the recall achievable by a fully supervised 2D object detector trained on target objects. we cannot compute Mean Average Precision (mAP), because our localization network follows the semantic segmentation rather than the object detection paradigm. As an alternative, we chose a confidence threshold for each method on each dataset separately, maximizing the F1 score. We then report precision and recall that correspond to the optimal threshold as well as the highest possible recall achieved by the detector. Table 1 summarizes the findings. As expected, YOLO performs better than the proposed method and OS2D, both in terms of precision and recall. Our method - not trained on the test objects - is outperformed by YOLO by 10%-20% in terms of best recall. At the same time, its recall is very close to the recall of YOLO with the confidence threshold corresponding to the best F1 score. Performance of OS2D is considerably worse, especially on more challenging datasets with more occlusion, e.g. LMO, HBD and YCB-V. Additionally, our method is ca. 800× faster with a runtime of only 25 milliseconds per object compared to 20 seconds per object for OS2D. This shows the advantages of the proposed method compared to state of the art in 2D one-shot detection.

Table 2. Percentages of correctly estimated poses w.r.t. the ADD on the Linemod [11] dataset for methods trained on synthetic data. All methods apart from ours, PPF and PfS require prior training on RGB target objects.

Modality	Method	Refinement	ADD	Time (ms)
RGB	DPOD [59]	DL [59]	54.2	-
	Ours	Mult. Hyp.	43.6	1343
	DPOD [59]	-	40.5	36
	OURS	-	39.3	96
	SSD6D [21]	DL [30]	34.1	-
	AAE [52]	-	31.4	24
	PfS [57]	-	22.5	-
	SSD6D [21]	-	9.1	-
RGBD	SSD6D [21]	ICP	90.9	100
	Ours	Mult. Hyp. + ICP	81.9	749
	Ours	Mult. Hyp.	80.1	722
	PPF [7]	ICP	78.8	-
	Ours	ICP	76.8	68
	Ours	-	73.3	60
	AAE [52]	ICP	71.5	224

4.2. 6 DoF Pose Results

In this section, we assess the accuracy of poses estimated using our one-shot method. However, due to a lack of relevant work, it is not a straightforward task. Geometry-based deep learning methods [38, 39] are evaluated on a single dataset using an old pose metric that the new state of the art methods do not report. Although Multi-Path AAE [51] claims to be one-shot, it employs a 2D object detector trained on target objects. We, however, still compare to Multi-Path AAE, as it serves as a state of the art upper bound of what pose accuracy is achievable with deep learning methods capable of estimating the pose of novel objects not seen during training. when depth data is available, PPF results are reported, because it is another truly one-shot method that does not require training on target objects. Other methods listed in the tables are explicitly trained on synthetic renderings of target objects, giving them an advantage over our method in terms of pose scores. Therefore, the results of our methods should not be directly compared to them; instead they should be used as a reference for what the standard 6 DoF pose estimation methods achieve. To summarize, we mainly compare our method to Multi-Path AAE on RGB images and to PPF on RGBD images.

In Table 2, the quality of pose estimation is reported using the ADD score for the Linemod dataset. We compare our approach to other methods that use only synthetic training data because they represent what can be accomplished without access to the training data from the target domain. Our method predicts very good poses despite not having been trained on Linemod objects. It significantly outperforms SSD6D [21], and outperforms AAE [52] and SSD6D with deep learning-based refinement [30], while

Table 3. Results on the Occlusion dataset [2] reported according to the Average Recall (AR) metric of the BOP challenge [14] on the BOP challenge subset of test images. All methods apart from ours and PPF [7] require prior training on target objects.

Method	Train data	Refinement	AR	Time (s)
CosyPose [22]	PBR	-	0.633	0.550
CDPN [27]		-	0.569	0.279
EPOS [12]		-	0.547	0.468
Pix2Pose [12]		-	0.363	1.310
Pix2Pose [12]		-	0.281	1.157
SSD6D [21]		-	0.139	-
EPOS [12]	synt	-	0.443	0.487
DPOD [59]		-	0.169	0.172
AAE [52]	mix	ICP	0.237	1.197
Multi-Path AAE [51]		-	0.217	0.200
AAE [52]		-	0.146	0.201
Drost, PPF [7]	-	ICP	0.527	15.947
Drost, PPF [7]		ICP, 3D edges	0.492	3.389
Ours + Kabsch		Mult. Hyp. + ICP	0.482	5.440
Ours + Kabsch		Mult. Hyp.	0.462	5.355
Ours + Kabsch		ICP	0.432	0.560
Ours + Kabsch		-	0.393	0.475
Ours + PnP		Mult. Hyp.	0.312	12.180
Ours + PnP		-	0.274	0.766

Table 4. Results on the Homebrewed dataset [20] reported according to the Average Recall (AR) metric of the BOP challenge [14] on the BOP challenge subset of test images. All methods apart from ours and PPF [7] require prior training on target objects.

Method	Train data	Refinement	AR	Time (s)
CDPNv2 [27]	PBR	-	0.722	0.273
CosyPose [22]		ICP	0.712	5.326
CDPNv2 [27]		ICP	0.712	0.713
Pix2Pose [36]		ICP	0.695	3.248
CosyPose [22]		-	0.656	0.417
EPOS [12]		-	0.58	0.657
Pix2Pose [36]		-	0.446	0.982
AAE [52]	synt	ICP	0.506	1.352
CDPN [27]		-	0.47	0.311
AAE [52]		-	0.346	0.19
Multi-Path AAE [51]		-	0.293	0.191
DPOD [59]		-	0.286	0.18
Drost, PPF [7]	-	ICP	0.671	144.029
Ours + Kabsch		Mult. Hyp. + ICP	0.605	4.508
Drost, PPF [7]		ICP	0.603	1.659
Ours + Kabsch		ICP	0.581	0.438
Ours + Kabsch		Mult. Hyp.	0.579	4.384
Ours + Kabsch		-	0.56	0.314
Ours + PnP		Mult. Hyp.	0.492	8.183
Ours + PnP		-	0.464	0.503

falling short of DPOD [59] by around 1%. Our method considerably outperform Pose from Shape [57] (PfS), which is another one-shot pose estimation method, even though PfS uses ground truth 2D detections and ground truth translations while estimating only rotation. Pose hypothesis verification raises the results by relative 10% from 39.3 to 43.6. If depth data is available, the pose can be estimated directly using 3D-3D correspondences and the Kabsch algorithm. This nearly doubles our method's ADD score, putting it above AAE with ICP refinement. Further pose hypothesis verification improves the results by around relative 10% putting it just above PPF. Segmentation of a single object takes 25 milliseconds, initial rotation approximation 10 milliseconds, 2D-2D matching 11 milliseconds. PnP takes 50 milliseconds on average, which makes the RGB-only pose estimation pipeline perform at approximately 10 FPS. If the Kabsch algorithm is used, the detector achieves 16 FPS. Additional ICP refinement on top of Kabsch slows down the detector to 14 FPS, which is still faster than other methods with ICP refinement. Pose hypothesis generation and verification in RGB takes around 1300 milliseconds and 550 milliseconds in depth images.

The results of our proposed method on the Occlusion (Table 3) and Homebrewed (Table 4) datasets show that it performs similarly to some of the methods that are trained on target objects with full supervision and trained separately for each object and scene. Even without refinement, our method outperforms Multi-Path AAE on both datasets. If depth data is available, our method falls short behind PPF only by a narrow margin while being an faster than the best performing PPF variant. On the YCB (Table 5) dataset, Multi-Path AAE outperforms the proposed method (Our+PnP), but it is important to note that methods trained on real or mixed data perform considerably better than methods trained solely on synthetic data on this dataset. On the other hand, our method outperforms PPF by a large margin. The results on all datasets clearly show the competitive quality of object detection and pose estimation in the proposed one-shot method and demonstrate that it generalizes well to objects, which were not seen during training. Moreover, its performance matches the performance of some of the previous state of the art methods even though they were trained on the target objects.

4.3. Ablation Studies

We conducted three main ablation studies to determine what factors contribute to the performance of our pipeline. Table 6 examines the architecture choices for the localization network, Table 7 analyses the pose estimation, while Figure 6 demonstrates the robustness of the method to a smaller number of templates and to larger angular errors between the ground truth and the matched templates.

Table 6 shows that if we only use the feature correlation as in OS2D [35,43], the network performs the worst achieving only 54% recall and 51% IoU. Only pixel-wise attention, on the other hand, yields comparable results. When both correlation and attention are used, network performance significantly increases which proves the effectiveness of the proposed architectural changes.

Table 7 snows an analysis of the impact of various stages on pose estimation. We start by replacing the first two stages of the pipeline with ground truth and then gradually replace it with actual predictions form our networks. ADD10 score is computed w.r.t. the number of correctly detected objects

Table 5. Results on the YCB-V dataset [56] reported according to the Average Recall (AR) metric of the BOP challenge [14] on the BOP challenge subset of test images. All methods apart from ours and PPF [7] require prior training on target objects.

Method	Train data	Refinement	AR	Time (s)
CDPNv2 [27]	PBR	ICP	0.532	1.034
EPOS [12]		-	0.499	0.764
CDPNv2 [27]		-	0.39	0.448
CosyPose [22]		-	0.574	0.342
EPOS [12]	synt	-	0.696	0.572
CDPN [27]		-	0.422	0.295
DPOD [59]		-	0.222	0.341
CosyPose [22]	mix	ICP	0.861	2.736
CosyPose [22]		-	0.821	0.241
Pix2Pose [36]		ICP	0.78	2.59
CDPNv2 [27]		-	0.532	0.143
AAE [52]		ICP	0.505	1.581
AAE [52]		-	0.377	0.179
Multi-Path AAE [51]		-	0.289	0.181
Ours + Kabsch	-	Mult. Hyp. + ICP	0.572	2.606
Ours + Kabsch		ICP	0.565	0.302
Ours + Kabsch		Mult. Hyp.	0.542	2.571
Ours + Kabsch		-	0.529	0.267
Drost, PPF [7]		ICP, 3D edges	0.344	6.27
Ours + PnP		Mult. Hyp.	0.332	5.389
Drost, PPF [7]		ICP, 3D edges	0.33	1.282
Ours + PnP		-	0.296	0.41

unless specified otherwise in "Recall", which multiplies the ADD score by the recall. The first line sets the upper bound for the pose estimation performance by effectively replacing the first two stages with ground truth and only using predictions from the 2D-2D matching network and PnP. The second line introduces the template matching. The overall score decreases by only 2% indicating that the second stage network performs precise template matching given the GT segmentation masks and that the 2D-2D matching network is robust to larger angular differences between poses in the object patch and in the template. A large performance drop is observed in the third line, when GT segmentation is replaced with the predicted segmentation. This is further emphasized in the last row, where the ADD10 score is corrected by detector's recall. These results indicate that the main error comes from erroneous initial viewpoint estimation. Furthermore, improving the predicted segmentation can further improve the overall performance of the one-shot pipeline by improving 2D recall and template matching. The table also shows that pose hypothesis verification helps filter out some of the erroneous poses and push the ADD score closer to the theoretical maximum of the method.

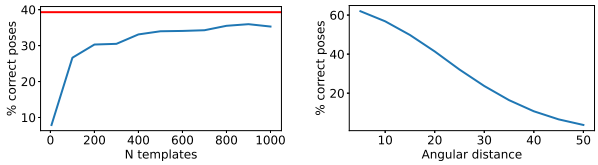
We followed the example of Multi-Path AAE and used 90K templates for all experiments. Figure 6a shows the ADD score on Linemod if 5 to 1K templates randomly sampled from the full set are used. The red line corresponds to the ADD score of 39.3 with all 90K templates. OSOP reaches ADD of 35.3 with only 1K templates and 38.6 with 5K templates. Figure 6b shows how the angular distance between the gt rotation and the matched template affects the ADD score. OSOP requires a smaller number of templates and can estimate poses even from more distant template matches because of dense correspondence estimation.

Table 6. Object localization and segmentation for various configurations of the proposed localization network on Linemod [11].

Configuration		Precision	Recall	IoU
Correlations	Attention			
✓		0.28	0.54	0.51
	✓	0.34	0.61	0.55
✓	✓	0.47	0.86	0.72

Table 7. ADD10 score on the Linemod [11] for different pipeline components.

Configuration						ADD10
GT segm.	Pred. Segm.	Closest tmpl.	Matched tmpl.	Recall	Mult. Hyp.	
+		+				60.9
+			+			58.9
	+		+			45.7
	+		+	+		39.3
	+		+		+	51.0
	+		+	+	+	43.6



(a) (b)

Figure 6. Impact of the number of templates (a) and the angular distance between the ground truth and the matched template (b) on the final ADD score on Linemod dataset [11].

5. Limitations

The method is predicated on the assumption that a pretrained feature extractor computes distinctive features from the synthetic rendering of the object and the input image, and that the features have higher correlations for templates and images that represent the object in similar poses. This assumption is influenced by the domain gap between real and synthetic images. This problem could potentially be remedied by unsupervised domain adaptation techniques.

6. Conclusion

We proposed a novel object detection and 6 DoF pose estimation method that generalizes well to the objects unseen during training. To the best of our knowledge, it is the first one shot object detection and pose estimation method that does not impose any specific requirements for the objects. Our novel neural network architecture for one shot object localization performs significantly better and faster than an alternative 2D one shot detector. Our evaluation on Linemod, Occlusion, Homebrewed, YCB and TLESS datasets for pose estimation demonstrates the effectiveness of our method, which achieves similar results to methods trained on synthetic data. The proposed pipeline allows for a considerable reduction of time needed to prepare training data and to train the model, as these steps are not needed for new unseen objects.

References

[1] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, 1992. 2

[2] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6d object pose estimation using 3d object coordinates. In *ECCV*, 2014. 1, 5, 7

[3] Mai Bui, Sergey Zakharov, Shadi Albarqouni, Slobodan Ilic, and Nassir Navab. When regression meets manifold learning for object recognition and pose estimation. In *ICRA*, 2018. 4

[4] Benjamin Busam, Hyun Jun Jung, and Nassir Navab. I like to move it: 6d pose estimation as an action decision process. *arXiv preprint arXiv:2009.12678*, 2020. 1

[5] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation. In *CVPR*, 2017. 3

[6] Xu Chen, Zijian Dong, Jie Song, Andreas Geiger, and Otmar Hilliges. Category level object pose estimation via neural analysis-by-synthesis. 2020. 1, 2

[7] Bertram Drost, Markus Ulrich, Nassir Navab, and Slobodan Ilic. Model globally, match locally: Efficient and robust 3d object recognition. In *2010 IEEE computer society conference on computer vision and pattern recognition*, 2010. 2, 6, 7, 8

[8] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 1981. 2

[9] Ross Girshick. Fast r-cnn. In *ICCV*, 2015. 3

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 3

[11] 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*, 2012. 5, 6, 8

[12] Tomas Hodan, Daniel Barath, and Jiri Matas. Epos: estimating 6d pose of objects with symmetries. In *CVPR*, 2020. 1, 2, 7, 8

[13] Tomas Hodan, Pavel Haluza, Stepan Obdrzalek, Jiri Matas, Manolis Lourakis, and Xenophon Zabulis. T-less: An rgb-d dataset for 6d pose estimation of texture-less objects. In *WACV*, 2017. 5

[14] Tomas Hodan and Antonin Melenovsky. Bop: Benchmark for 6d object pose estimation: https://bop.felk.cvut.cz/home/, 2019. 1, 2, 5, 6, 7, 8

[15] Ting-I Hsieh, Yi-Chen Lo, Hwann-Tzong Chen, and Tyng-Luh Liu. One-shot object detection with co-attention and co-excitation. In *NeurIPS*. 2019. 3

[16] Omid Hosseini Jafari, Siva Karthik Mustikovela, Karl Pertsch, Eric Brachmann, and Carsten Rother. ipose: instance-aware 6d pose estimation of partly occluded objects. In *ACCV*, 2018. 1, 2

[17] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In *CVPR*, 2019. 3

[18] Leonid Karlinsky, Joseph Shtok, Sivan Harary, Eli Schwartz, Amit Aides, Rogerio Feris, Raja Giryes, and Alex M Bronstein. Repmet: Representative-based metric learning for classification and few-shot object detection. In *CVPR*, 2019. 3

[19] Roman Kaskman, Ivan Shugurov, Sergey Zakharov, and Slobodan Ilic. 6 dof pose estimation of textureless objects from multiple rgb frames. In *ECCV Workshops*, 2020. 1, 2

[20] Roman Kaskman, Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Homebreweddb: Rgb-d dataset for 6d pose estimation of 3d objects. In *ICCVW*, 2019. 1, 5, 6, 7

[21] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *ICCV*, 2017. 1, 6, 7

[22] Y. Labbe, J. Carpentier, M. Aubry, and J. Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *ECCV*, 2020. 1, 2, 7, 8

[23] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *IJCV*, 2009. 2

[24] Fu Li, Ivan Shugurov, Benjamin Busam, Minglong Li, Shaowu Yang, and Slobodan Ilic. Polarmesh: A star-convex 3d shape approximation for object pose estimation. *IEEE Robotics and Automation Letters*, 2022. 1, 2

[25] Fu Li, Ivan Shugurov, Benjamin Busam, Shaowu Yang, and Slobodan Ilic. Ws-ope: Weakly supervised 6-d object pose regression using relative multi-camera pose constraints. *IEEE Robotics and Automation Letters*, 2022. 1

[26] Fu Li, Hao Yu, Ivan Shugurov, Benjamin Busam, Shaowu Yang, and Slobodan Ilic. Nerf-pose: A first-reconstruct-then-regress approach for weakly-supervised 6d object pose estimation. *arXiv preprint arXiv:2203.04802*, 2022. 1

[27] Zhigang Li, Gu Wang, and Xiangyang Ji. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *ICCV*, 2019. 1, 2, 7, 8

[28] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 3

[29] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 3

[30] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. Deep model-based 6d pose refinement in rgb. In *ECCV*, 2018. 6

[31] Fabian Manhardt, Gu Wang, Benjamin Busam, Manuel Nickel, Sven Meier, Luca Minciullo, Xiangyang Ji, and Nassir Navab. Cps++: Improving class-level 6d pose and shape estimationfrom monocular images with self-supervised learning. *arXiv preprint arXiv:2003.05848v3*, 2020. 1, 2

[32] Claudio Michaelis, Ivan Ustyuzhaninov, Matthias Bethge, and Alexander S Ecker. One-shot instance segmentation. *arXiv*, 2018. 3

[33] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *3DV*, 2016. 4

[34] MVTec. Bop: Benchmark for 6d object pose estimation: https://www.mvtec.com/products/halcon/, 2019. 2

[35] Anton Osokin, Denis Sumin, and Vasily Lomakin. Os2d: One-stage one-shot object detection by matching anchor features. In *ECCV*, 2020. 2, 3, 6, 7

[36] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *ICCV*, 2019. 1, 2, 7, 8

[37] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *CVPR*, 2019. 1, 2

[38] Giorgia Pitteri, Aurélie Bugeau, Slobodan Ilic, and Vincent Lepetit. 3d object detection and pose estimation of unseen objects in color images with local surface embeddings. In *ACCV*, 2020. 1, 2, 6

[39] Giorgia Pitteri, Slobodan Ilic, and Vincent Lepetit. Cornet: generic 3d corners for 6d pose estimation of new objects without retraining. In *ICCV Workshops*, 2019. 1, 2, 6

[40] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 3

[41] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv*, 2018. 6

[42] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *TPAMI*, 2016. 3

[43] Ignacio Rocco, Relja Arandjelovic, and Josef Sivic. Convolutional neural network architecture for geometric matching. In *CVPR*, 2017. 3, 7

[44] I. Rocco, R. Arandjelović, and J. Sivic. Convolutional neural network architecture for geometric matching. *TPAMI*, 2018. 3

[45] Ignacio Rocco, Relja Arandjelović, and Josef Sivic. End-to-end weakly-supervised semantic alignment. In *CVPR*, 2018. 3

[46] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 2015. 4

[47] I. Shugurov, I. Pavlov, S. Zakharov, and S. Ilic. Multi-view object pose refinement with differentiable renderer. *IEEE Robotics and Automation Letters*, 2021. 1

[48] Ivan Shugurov, Ivan Pavlov, Sergey Zakharov, and Slobodan Ilic. Multi-view object pose refinement with differentiable renderer. *IEEE Robotics and Automation Letters*, 2021. 2

[49] Ivan Shugurov, Sergey Zakharov, and Slobodan Ilic. Dpodv2: Dense correspondence-based 6 dof pose estimation. *TPAMI*, 2021. 1, 2

[50] Chen Song, Jiaru Song, and Qixing Huang. Hybridpose: 6d object pose estimation under hybrid representations. In *ICCV*, 2020. 1, 2

[51] Martin Sundermeyer, Maximilian Durner, En Yen Puang, Zoltan-Csaba Marton, Narunas Vaskevicius, Kai O Arras, and Rudolph Triebel. Multi-path learning for object pose estimation across domains. In *CVPR*, 2020. 1, 2, 4, 6, 7, 8

[52] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *ECCV*, 2018. 1, 6, 7, 8

[53] Bugra Tekin, Sudipta N Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *CVPR*, 2018. 2

[54] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, 2019. 3

[55] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *CVPR*, 2019. 1, 2

[56] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. 2018. 5, 8

[57] Yang Xiao, Xuchong Qiu, Pierre-Alain Langlois, Mathieu Aubry, and Renaud Marlet. Pose from shape: Deep pose estimation for arbitrary 3d objects. *arXiv preprint arXiv:1906.05105*, 2019. 1, 3, 6, 7

[58] Sergey Zakharov, Wadim Kehl, Benjamin Planche, Andreas Hutter, and Slobodan Ilic. 3d object instance recognition and pose estimation using triplet loss with dynamic margin. In *IROS*, 2017. 4, 5

[59] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *ICCV*, 2019. 1, 2, 6, 7, 8