NeuSurfEmb: A Complete Pipeline for Dense Correspondence-based 6D Object Pose Estimation without CAD Models

Francesco Milano[1], Jen Jen Chung[2], Hermann Blum[1], Roland Siegwart[1], Lionel Ott[1]

Abstract— **State-of-the-art approaches for 6D object pose estimation assume the availability of CAD models and require the user to manually set up physically-based rendering (PBR) pipelines for synthetic training data generation. Both factors limit the application of these methods in real-world scenarios. In this work, we present a pipeline that does not require CAD models and allows training a state-of-the-art pose estimator requiring only a small set of real images as input. Our method is based on a NeuS2 [1] object representation, that we learn through a semi-automated procedure based on Structure-from-Motion (SfM) and object-agnostic segmentation. We exploit the novel-view synthesis ability of NeuS2 and simple *cut-and-paste* augmentation to automatically generate photorealistic object renderings, which we use to train the correspondence-based SurfEmb [2] pose estimator. We evaluate our method on the LINEMOD-Occlusion dataset, extensively studying the impact of its individual components and showing competitive performance with respect to approaches based on CAD models and PBR data. We additionally demonstrate the ease of use and effectiveness of our pipeline on self-collected real-world objects, showing that our method outperforms state-of-the-art CAD-model-free approaches, with better accuracy and robustness to mild occlusions. To allow the robotics community to benefit from this system, we will publicly release it at https://www.github.com/ethz-asl/neusurfemb.**

I. Introduction

Estimating the 6D pose of objects from image observations is a long-standing problem in computer vision and of broad interest to several important real-world applications, including robotic manipulation [3]–[5], augmented reality [6]–[8], and object-level mapping [9], [10].

Many of the current state-of-the-art approaches require a high-fidelity, often textured, CAD model of an object to estimate its pose [2], [11]–[14]. While the datasets used in recent evaluation benchmarks [15]–[17] provide this information, in practical real-world applications, obtaining an accurate, textured CAD reconstruction is often non-trivial, usually requiring manual design or specialized equipment for data collection or intensive post-processing [16], [18]. Moreover, the vast majority of the proposed approaches are trained on large synthetic datasets generated through physically-based rendering (PBR) pipelines [19], [20]. These pipelines produce photorealistic images with physically accurate modelling of light and material properties; however, they require textured CAD models and proper setup and parameter configuration from an experienced user, which makes their application to a new, real-world object non-straightforward.

[1]ETH Zurich, Switzerland, [2]The University of Queensland, Australia.

This work has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 101017008.

To be of practical use for a real-world system, *e.g.*, a robot or an augmented reality headset, a pose estimation algorithm would instead ideally require the user to provide only a small set of observations of an object of interest. With this goal in mind, a number of *model-free* approaches for 6D pose estimation have been proposed, which typically construct a Structure-from-Motion (SfM)-based model of the object and later relocalize the camera with respect to it [21], [22]. While these methods allow relaxing the assumption of a CAD model, they tend to be less accurate than state-of-the-art methods leveraging CAD models and PBR data, and typically show limited robustness to occlusions.

In this work, we propose a framework that allows training a pose estimator for real-world objects without requiring a CAD model or a PBR synthetic dataset, while still achieving performance comparable to state-of-the-art approaches that require the latter. Our method is provided in the form of a semi-automated pipeline that simply requires a sparse set of image observations of the object and a bounding box indicating the object of interest in one frame. After training, our system allows estimating the 6D pose of the object from a single RGB image, with optional depth-based refinement. We use a neural implicit surface reconstruction method (NeuS2 [1]) as the underlying representation for the object of interest. Using SfM in combination with state-of-the-art object-agnostic segmentation [23] and tracking [24], we automatically estimate poses and object masks for each of the reference images. We then use these frames to train an object-level NeuS2, which compactly and accurately reconstructs the object, effectively replacing a CAD model. At the same time, we show that NeuS2 can replace a more involved PBR pipeline and efficiently generate renderings to train a pose estimator. For the latter, we leverage SurfEmb [2], a recent method based on dense correspondences.

We evaluate our approach on the LINEMOD-Occlusion [25] dataset, and conduct extensive ablations to highlight the effect of each component in our pipeline. We additionally demonstrate our method on a set of real-world objects, performing both qualitative and quantitative evaluations against state-of-the-art baselines that, like our method, are applicable when no CAD models are available. Our pipeline, which we name NeuSurfEmb, achieves comparable performance to CAD-model-based methods and outperforms previous CAD-model-free approaches.

In summary, our main contributions are the following:

i. A pipeline for 6D object pose estimation requiring only a small set of real RGB images as input.
ii. Extensive ablation studies on our object representation,

training data, and other components of our pipeline.

iii. Evaluation on a standard dataset and on real-world data, showing that our approach achieves competitive performance against state-of-the-art methods, while being applicable to real-world objects.

iv. An open-source implementation to easily train and deploy our pipeline for novel objects.

II. Related work

A. CAD model-based object pose estimation

A large number of state-of-the-art approaches for 6D object pose estimation rely on the assumption that a CAD model of the object of interest is available. On one side, the CAD model is used to generate synthetic data through photorealistic rendering pipelines, often based on PBR [20], [26]. For this reason, high-fidelity object texture is needed to produce high-quality renderings with limited domain gap with respect to the real data on which the trained algorithms are evaluated. On the other side, the CAD model is used during the training of the pose estimation algorithm. For instance, keypoint-based methods [27]–[29] predict the 2D location of pre-defined salient points, and use the object model to estimate the object pose based on 2D-3D correspondences. Coordinate-based methods [12], [14], [30] predict a 3D coordinate, defined according to the object model, for each pixel in the input image. [13] and [31] render reference images from the textured CAD model, use pre-trained networks to find the reference image that best matches the test one, and subsequently refine the pose. Correspondence-based methods [2], [32], which achieve state-of-the-art robustness to occlusions, compute *dense* correspondences between image pixels and 3D points on the object model. We base our pose estimation algorithm on SurfEmb [2], a recent method from the latter category, and relax its assumptions of a CAD model and PBR synthetic dataset.

B. CAD model-free object pose estimation

SfM-based methods are the current state of the art for CAD model-free object pose estimation [21], [22]. These methods assume a set of reference images, which are used to construct a sparse [21] or semi-dense [22] point cloud, using SfM. Together with the reference images, the point cloud acts as a 3D model, and is used to estimate the pose of the object in a test image through feature matching and PnP. SfM-based methods tend to be less accurate and robust to occlusions than state-of-the-art, CAD-model-based methods, but can easily be applied in a real-world scenario where no CAD models are available. We follow a similar setup in our method, assuming a given set of reference images and running SfM on them; however, we base our object model on NeuS2 rather than a point cloud.

C. Object pose estimation via neural implicit representations

Similarly to our method, the recent BundleSDF [33] and TexPose [34] perform 6D pose estimation based on a neural implicit object representation. However, BundleSDF additionally requires depth images as input, while TexPose assumes a CAD model and a PBR synthetic dataset.

III. Method

An overview of our method is shown in Fig. 1. We first present the steps to generate an object model and a synthesized dataset based on NeuS2 (yellow boxes), then we detail the steps to learn 2D-3D correspondences (green box), and finally we describe the procedure to estimate the pose of the object of interest in a test image (purple box).

A. NeuS2-based object model and dataset

Similarly to other CAD-model-free methods [21], [22], in our setting we assume to have available a small set of N images $\{\mathbf{I}_i\}$ (with $N \approx 100$), captured at roughly uniformly-distributed viewpoints around the object. These images are used to construct a reference object model with respect to which the object pose in test images is later estimated. We perform COLMAP [36]-based SfM to retrieve camera poses $\{\mathbf{P}_i\}$ associated to the reference images $\{\mathbf{I}_i\}$. However, in contrast to the sparse or semi-dense cloud of triangulated points that form the SfM-based object models in [21], [22], we learn a dense object model based on NeuS2 [1].

To this end, we first extract object masks from the reference images using a semi-automatic pipeline based on Segment Anything (SAM) [23] and MixFormer [24]. We assume that the object of interest is not occluded in the reference images, that the latter are extracted from a temporal sequence, and that a bounding box $\mathcal{B}_1$ around the object in the first frame $\mathbf{I}_1$ is provided. $\mathcal{B}_1$ is used to prompt SAM for an object mask for $\mathbf{I}_1$. We then fit a tight bounding box $\hat{\mathcal{B}}_1$ around the predicted object mask $\tilde{\mathbf{I}}_1$, and provide it to MixFormer as initialization for the subsequent frame $\mathbf{I}_2$. The bounding box $\hat{\mathcal{B}}_2^{\text{tr}}$ returned as output by MixFormer is then used as prompt for SAM to extract a mask $\tilde{\mathbf{I}}_2$ for the image $\mathbf{I}_2$, and the process is repeated for all the frames $\mathbf{I}_t$, with $t \in \{2, \ldots, N\}$, using $\hat{\mathcal{B}}_{t-1}$ as initialization for MixFormer and the tracked bounding box $\hat{\mathcal{B}}_t^{\text{tr}}$ as a prompt for SAM. We find that this process accurately segments the object in the majority of the frames, and we provide additional tools based on SAM to refine the few poorly segmented frames, for instance by allowing multiple prompts per frame.

We use the extracted masked images $\{\tilde{\mathbf{I}}_i\}$ and the estimated camera poses $\{\mathbf{P}_i\}$ to train an object-level NeuS2 model through inverse volume rendering, using its standard supervision setting that combines a robust color loss with a regularization Eikonal loss [1]. By relying on an underlying signed distance field (SDF) and employing an unbiased volume rendering formulation [37], NeuS2 is able to accurately reconstruct the object surface, which we extract either as a mesh model, using Marching Cubes [38], or as a point cloud, by rendering per-pixel 3D coordinates from different viewpoints and aggregating them.

At the same time, we exploit the ability of NeuS2 to synthesize novel high-fidelity views of the object to efficiently produce renderings $\{\mathbf{I}_i^{\text{syn}}\}$ of the object from camera poses $\{\mathbf{P}_i^{\text{syn}}\}$ that we randomly sample from the top hemisphere around the object. To ensure that the coordinate frame of the NeuS2 model, and consequently the rendered images, align with the natural frame of the object, we perform NeuS2

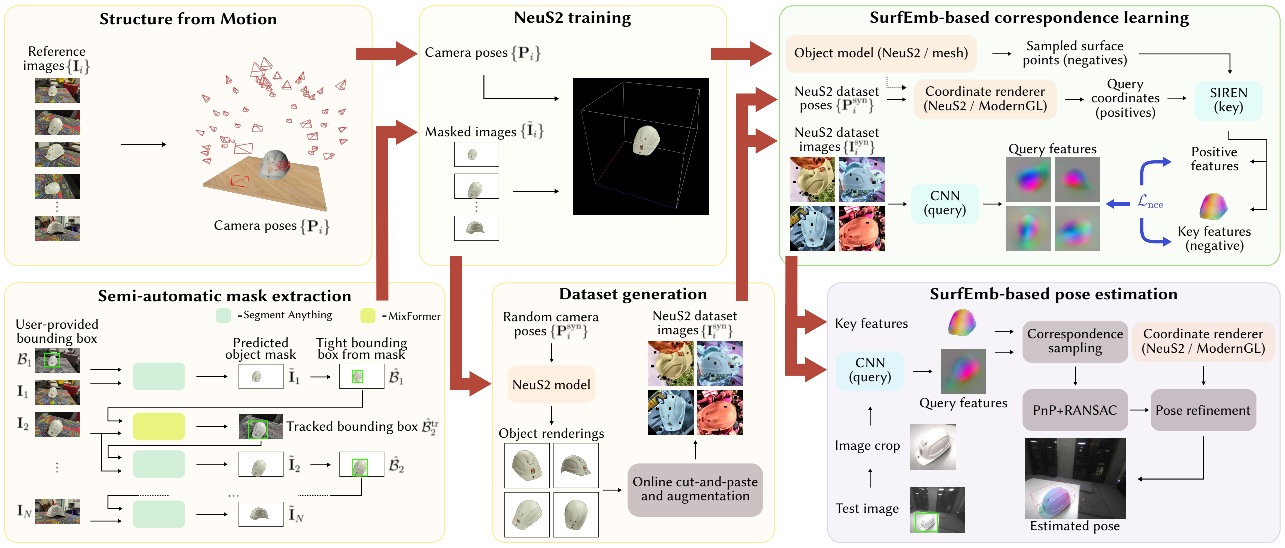
Structure from Motion
Reference images $\{\mathbf{I}_i\}$
Camera poses $\{\mathbf{P}_i\}$
NeuS2 training
Camera poses $\{\mathbf{P}_i\}$
Masked images $\{\tilde{\mathbf{I}}_i\}$
SurfEmb-based correspondence learning
Object model (NeuS2 / mesh)
Sampled surface points (negatives)
NeuS2 dataset poses $\{\mathbf{P}_i^{\text{syn}}\}$
Coordinate renderer (NeuS2 / ModernGL)
Query coordinates (positives)
SIREN (key)
NeuS2 dataset images $\{\mathbf{I}_i^{\text{syn}}\}$
Query features
Positive features
CNN (query)
$\mathcal{L}_{\text{nce}}$
Key features (negative)
Semi-automatic mask extraction
=Segment Anything
=MixFormer
User-provided bounding box
$\mathcal{B}_1$
$\mathbf{I}_1$
$\mathbf{I}_2$
$\mathbf{I}_N$
Predicted object mask
Tight bounding box from mask
$\tilde{\mathbf{I}}_1$
$\tilde{\mathcal{B}}_1$
Tracked bounding box $\tilde{\mathcal{B}}_2^{\text{tr}}$
$\tilde{\mathbf{I}}_2$
$\tilde{\mathcal{B}}_2$
$\tilde{\mathbf{I}}_N$
Dataset generation
Random camera poses $\{\mathbf{P}_i^{\text{syn}}\}$
NeuS2 dataset images $\{\mathbf{I}_i^{\text{syn}}\}$
NeuS2 model
Object renderings
Online cut-and-paste and augmentation
SurfEmb-based pose estimation
Key features
Correspondence sampling
Coordinate renderer (NeuS2 / ModernGL)
CNN (query)
Query features
PnP+RANSAC
Pose refinement
Image crop
Test image
Estimated pose

Fig. 1. Overview of the proposed method. Starting from a set of reference images $\{\mathbf{I}_i\}$ around the object of interest, and using Structure-from-Motion and a pipeline based on Segment Anything [23] and the object tracker MixFormer [24] to estimate corresponding camera poses $\{\mathbf{P}_i\}$ and object masks $\{\tilde{\mathbf{I}}_i\}$, we construct an object model and synthesized dataset by training a NeuS2 [1] model and generating renderings from novel views $\{\mathbf{P}_i^{\text{syn}}\}$ (yellow boxes). We use the generated object model and synthesized dataset, augmented online using cut-and-paste [35] to simulate occlusions and background variations, to learn feature-based dense 2D-3D correspondences based on SurfEmb [2] (green box). We then estimate the object pose in a test image by sampling correspondences based on the learned object features and the predicted image features and using PnP with RANSAC and pose refinement (purple box).

training in two steps: (i) We re-orient the reference SfM camera poses $\{\mathbf{P}_i\}$ based on their viewing directions so that the NeuS2 coordinate frame roughly coincides with the center of the observed object; (ii) We extract a point cloud from the NeuS2 model trained in the first step and fit a 3D oriented bounding box to it. We redefine the NeuS2 coordinate frame to be centered in the bounding box and aligned with its axes. We re-align the reference camera poses accordingly, and then re-train the NeuS2 model with the new coordinate frame. Since the subsequent training steps require the images to be cropped at a fixed square resolution around the object, to maintain efficiency and image quality we directly render the synthesized images $\{\mathbf{I}_i^{\text{syn}}\}$ cropped around the object. For each viewpoint $\mathbf{P}_i^{\text{syn}}$, we do so by reprojecting the object point cloud onto the image plane, fitting a 2D bounding box around it and adapting the camera intrinsics to render only within the bounding box.

B. Correspondence training

We use the NeuS2-based object model and dataset to learn 2D-3D correspondences using SurfEmb [2]. In particular, we instantiate a convolutional neural network (CNN) – also referred to as *query network* – and a coordinate field based on SIREN [39] – *key network*, to respectively return a high-dimensional feature $\mathbf{f}_q(\mathbf{p}) \in \mathbb{R}^d$ for each pixel $\mathbf{p}$ in the input image and a feature vector $\mathbf{f}_k(\mathbf{x}) \in \mathbb{R}^d$ for each 3D point $\mathbf{x}$ on the object surface. For each input image, we additionally render 3D coordinates corresponding to each pixel, either by using a mesh-based ModernGL [40] renderer (as in SurfEmb), that we provide with the Marching Cubes mesh extracted from NeuS2, or by directly rendering the coordinates using NeuS2. We then query the key network at the rendered 3D coordinates and train the two networks using a contrastive Info-NCE loss $\mathcal{L}_{\text{nce}}$ [41]. Like SurfEmb, we additionally output from the query network an object mask, supervised with a cross-entropy loss with respect to the ground-truth object mask.

Crucially, since the rendered images $\{\mathbf{I}_i^{\text{syn}}\}$ only contain the object of interest, we apply online *cut-and-paste* [35] augmentation to the foreground and background of each training image, to simulate occlusions and background variations, respectively. For the background, we use, with equal probability, random noise and a crop of an image sampled from the PASCAL-VOC dataset [42]; for the foreground, we randomly select an instance from a PASCAL-VOC image and place it on the rendered image so that the percentage of occluded object pixels varies between 20% and 70%. We apply extensive color augmentation and in-plane affine transformations, as done in SurfEmb, and additionally employ white-balancing augmentation using the method of [43], which we find beneficial for the generalization of the method.

C. Pose estimation

Given a test image containing the object of interest, we estimate the 6D pose of the object with respect to the camera using correspondence-based method of [2]. In particular, assuming a 2D bounding box provided by an external object detector, we crop and rescale the input image to the resolution used during training and feed it to the query network. We then compute the similarity between the output query features, weighted by the predicted object mask, and the features returned by the key network for a uniform set of surface points. The resulting similarity matrix is used to sample 2D-3D correspondences using importance sampling. A set of candidate poses is obtained from these correspondences using PnP+RANSAC, and a refinement step is applied to the best-scoring pose using a coordinate renderer (we refer the reader to Sec. 3.3 of [2] for exact details).

Similarly to the correspondence learning step, in our setup either a mesh-based renderer or NeuS2 can be used as coordinate renderer. Finally, an additional refinement step can be performed if a depth image is available.

IV. Experiments and Results

In the following, we assess our method's performance and the impact of its components. We cover evaluation metrics and training details in Sec. IV-A. We analyze the reconstruction quality of NeuS2 in Sec. IV-B, our pose estimation performance on LINEMOD-Occlusion in Sec. IV-C, and the impact of our NeuS2-based object model and dataset in Sec. IV-D. Lastly, in Sec. IV-E, we test our method on self-collected real-world data.

A. *Experimental setup*

For the experiments on LINEMOD-Occlusion, we report the standard BOP Average Recall error measure, $\mathrm{AR}_{\mathrm{BOP}}$, which takes into account both object symmetries and occlusions in the scene [19]. For the real-world experiments, we use the standard ADD(−S) [15], [44] and $5\,\mathrm{cm},\ 5^\circ$ [45] metrics for the scenes with no occlusions and $\mathrm{AR}_{\mathrm{BOP}}$ and $5\,\mathrm{cm},\ 5^\circ$ for the scenes with occlusions.

We train NeuS2 for 20 000 steps. For each object, we generate 10 000 images at a resolution of 224×224 pixels. For correspondence learning, we use the same network architectures as in SurfEmb, and train a model for each object for 50 epochs, to achieve a similar number of iterations as in the original method [2]. NeuS2 training takes 10 min, dataset generation 20 to 30 min, and correspondence learning 1 to 1.5 days, on a single NVIDIA RTX 2080 Ti GPU.

B. *NeuS2 reconstruction quality*

Given the correspondence-based nature of our pose estimation method, the geometric accuracy of the 3D object model might play an important role in the quality of the estimated pose, since selecting potentially inaccurate 3D surface points as samples for the PnP+RANSAC step might directly result in errors in the predicted pose. To investigate the impact of the model accuracy, we first assess the reconstruction quality of our NeuS2-based 3D models. We evaluate it for the 8 objects in the LINEMOD-Occlusion dataset [25] and report it as the forward Chamfer distance between the NeuS2 mesh reconstructions and the corresponding ground-truth CAD models available in the dataset. For each object, we train a NeuS2 model using the images and ground-truth poses and masks from the corresponding scene in the occlusion-free LINEMOD dataset [15]. As shown in Fig. 2, NeuS2 achieves very accurate reconstruction (in the order of 1 mm or lower reconstruction error) for 5 out of 8 objects. For the remaining 3 objects, a closer look at the reconstructed surfaces shows that two main failure modes can be highlighted (bottom row): 1. Partial holes inside the objects tend to not be properly captured (can, holepuncher); 2. Surface parts that are not visible in the training views cannot be reconstructed, as for instance the cone-like structures at the bottom of the eggbox. We investigate the impact of the reconstruction quality on the pose predictions in Sec. IV-D.

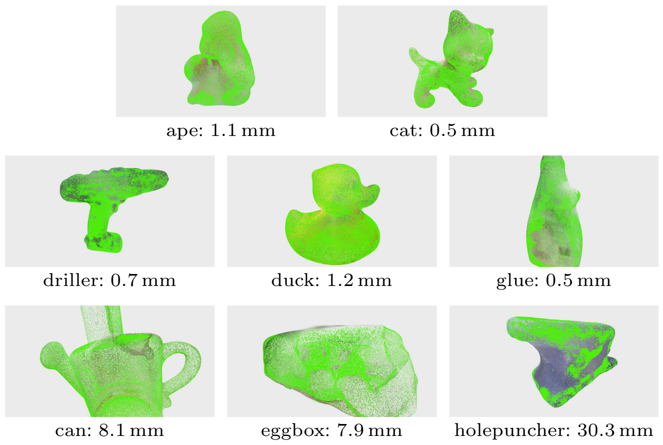

Fig. 2. Example NeuS2 reconstructions (shown as textured point cloud), overlaid on the point cloud sampled from the CAD model (shown in green) on the objects from LINEMOD-Occlusion. Next to the object names we report the forward Chamfer distance with respect to the CAD model.

Method	Training renderer	Training object model	Training images	$\mathrm{AR}_{\mathrm{BOP}}$ RGB	$\mathrm{AR}_{\mathrm{BOP}}$ RGB-D
NeuSurfEmb	NeuS2	NeuS2	NeuS2 (10k)	0.554	0.666
	GL-based	NeuS2	NeuS2 (10k)	0.570	0.681
	GL-based	NeuS2	PBR	0.646	0.752
	GL-based	CAD	NeuS2 (10k)	0.568	0.678
SurfEmb [2]	GL-based	CAD	PBR	**0.656**	**0.758**
EPOS [46]	-	CAD	PBR	0.547	-
CDPNv2 [12]	-	CAD	PBR	0.624	-
PVNet [29]	-	CAD	PBR	0.575	-
CosyPose [11]	-	CAD	PBR	0.633	0.714

TABLE I

Pose estimation performance averaged across the objects, LINEMOD-Occlusion.The baseline results are taken from [2].

C. *LINEMOD-Occlusion experiments*

We evaluate the pose estimation performance of our method on the LINEMOD-Occlusion dataset, which contains pose annotations for a subset of 8 objects from the original LINEMOD dataset and in which, unlike LINEMOD, the objects of interest present occlusions. For each object, we train a NeuS2 model following the same setup as in Sec. IV-B. Following the standard practice in the literature [2], we use the object detections from CosyPose [11] to crop the test images for pose estimation. We compare the performance of our method to state-of-the-art approaches which, unlike our method, assume a CAD model and a PBR synthetic dataset.

Table I shows the results of the evaluation. Both with RGB-only and with RGB-D inputs, NeuSurfEmb achieves comparable performance to several CAD-model-based baselines [29], [46] and is outperformed by a small margin by others [2], [11], [12]. While the coordinate renderer (cf. rows 1 and 2) and the object model (cf. rows 2 and 4) both have minimal impact on the output performance, which confirms that NeuS2 is able to accurately approximate the ground-truth object geometry, we find that the major differentiating factor is the type of images used for correspondence learning. When using PBR images instead of NeuS2-generated ones, our method achieves virtually the same performance as the leading method, SurfEmb (cf. rows 3 and 5). We hypothesize that this performance discrepancy is largely due to the way

Training images	Object model		$\mathrm{AR}_{\mathrm{BOP}}$								
	Training	Pose estimation	ape	can	cat	driller	duck	eggbox	glue	holepuncher	AVG Scene
NeuS2 (10k)	NeuS2	NeuS2	0.519	0.761	0.467	0.604	0.638	0.284	0.546	0.695	0.570
NeuS2 (10k)	NeuS2	CAD	0.527	0.751	0.468	0.604	0.652	0.251	0.562	0.438	0.534
PBR	NeuS2	NeuS2	0.627	0.790	0.609	0.806	0.633	0.409	0.635	0.622	0.646
PBR	NeuS2	CAD	0.613	0.776	0.621	0.805	0.650	0.291	0.647	0.465	0.610
NeuS2 (10k)	CAD	CAD	0.539	0.759	0.487	0.554	0.639	0.270	0.576	0.688	0.568
NeuS2 (10k)	CAD	NeuS2	0.534	0.709	0.486	0.549	0.619	0.290	0.554	0.622	0.549
PBR	CAD	CAD	0.593	0.788	0.625	0.783	0.623	0.424	0.584	0.695	0.646
PBR	CAD	NeuS2	0.594	0.792	0.601	0.787	0.612	0.514	0.600	0.642	0.648

TABLE II

EFFECT OF THE TRAINING IMAGES AND THE OBJECT MODEL IN POSE ESTIMATION, LINEMOD-OCCLUSION. RGB-ONLY INPUT.

Training object model	Training images	$\mathrm{AR}_{\mathrm{BOP}}$								
		ape	can	cat	driller	duck	eggbox	glue	holepuncher	AVG Scenes
NeuS2	NeuS2 (10k)	0.782	0.957	0.863	0.930	0.790	0.880	0.781	0.842	0.853
NeuS2	PBR	0.865	0.897	0.857	0.939	0.833	0.779	0.779	0.695	0.831
CAD	NeuS2 (10k)	0.788	0.948	0.871	0.909	0.797	0.943	0.822	0.845	0.865
CAD	PBR	0.804	0.901	0.875	0.954	0.828	0.780	0.705	0.780	0.829

TABLE III

POSE ESTIMATION PERFORMANCE ON THE LINEMOD DATASET, EVALUATED FOR THE 8 OBJECTS ALSO CONTAINED IN LINEMOD-OCCLUSION. ALL MODELS USE THE MODERNGL RENDERER AND ARE EVALUATED WITH RGB-ONLY INPUTS.

our image generation approach simulates occlusions, which appear less realistic compared to those in PBR images. We validate this hypothesis and further investigate the effect of both object model and synthesized images on the final performance in the next Section.

D. *Ablation: Effect of NeuS2-based object model and images*

In this ablation study, we report the performance for each object individually, to better capture potential object-specific factors. Table II shows the results of the evaluation, where in the top 4 rows we evaluate models trained with NeuS2 object models and in the bottom 4 rows those trained with CAD models. From the pairs of rows in Tab. II, *i.e.*, 1-2, 3-4, *etc.* we can see that for all the objects except eggbox, holepuncher, and to some extent can, using a different model for training and pose estimation has minimal effect on the pose estimation performance. This aligns with the results discussed in Sec. IV-B on the per-object reconstruction quality. In addition, it should be noted that similar performance is obtained across all objects when training and pose estimation both use the same model (Neus2 or CAD, *e.g.*, rows 1 and 5). We attribute this result to the fact that the key network learns to assign low-norm features to parts of the object that are never or only rarely observed during training, which as noted in Sec. IV-B constitute the main factor of geometric discrepancy between the two types of object model. As a result, 3D points on these parts are sampled only with low probability during pose estimation, yielding limited inconsistencies between the pose estimates for the two types of model. A second point that can be noted is that for both NeuS2 and CAD, the effect of the training images is largely dependent on the specific object. We hypothesize that this variability is due to a combination of differences in the object textures, which may be simulated more accurately by PBR for certain objects, and of the different effectiveness of how occlusions are simulated in the two types of images. To further investigate the impact of the simulated occlusions on the results, we test the models trained for the experiments in Sec. IV also on the occlusion-free LINEMOD dataset, reporting the performance for the 8 objects shared across both datasets. We use ground-truth bounding boxes to crop the test images. The results of this ablation, reported in Tab. III, show that when no occlusions are present in the test data, NeuS2-generated images perform on-par or at times even better than PBR images. This finding supports our hypothesis that our *cut-and-paste* strategy for occlusion simulation has margin for improvement, but at the same time indicates that our NeuS2-synthesized images are overall effective for training a pose estimator, in particular when very high robustness to occlusions is not the main requirement. Importantly, we stress that while for this particular ablation the domain of the images used to reconstruct the NeuS2 model and that of the test images coincide, during training our pose estimator is provided exclusively with *synthesized* images. Due to our *cut-and-paste* strategy, background and foreground of the synthesized images used in correspondence learning are significantly different from those of the test images, which together with our extensive data augmentation ensures that no overfitting to the test images can occur. To further validate this point and show that our method achieves good generalization, in the real-world experiments presented in the next Section we change lighting conditions, background, and camera characteristics between NeuS2 training and pose estimation.

E. *Real-world experiments*

To demonstrate the effectiveness and ease of use of our method for real-world applications, we collect recordings of 5 different objects (cf. Fig. 3) and run our pipeline as described in Sec. III, including obtaining camera poses and extracting object masks to train NeuS2. Note that the chosen objects capture different types of challenges, including low degree of texture (helmet, kettle), structural symmetries (bluebox, greybox), and complex geometry (extinguisher). We compare our method to two recent state-of-the-art approaches, Gen6D [21] and OnePose++ [22], both of which, similarly to our method, do not require a CAD model of the objects of interest. We note that CAD-model-free methods

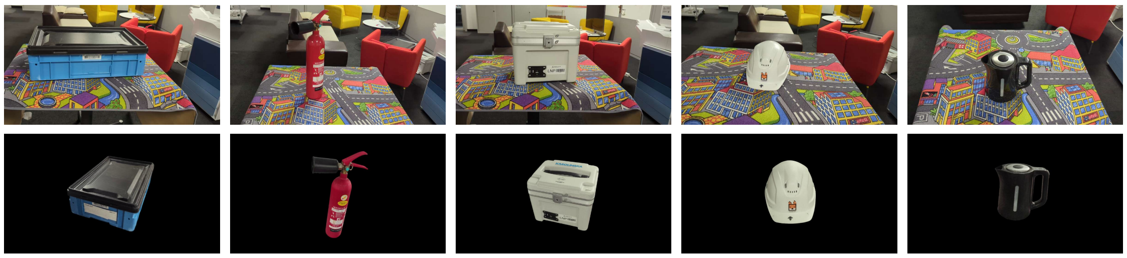

Fig. 3. Example images captured for model construction in the real-world experiments (top row) and corresponding NeuS2 reconstructions (bottom row). The objects depicted from left to right are: bluebox, extinguisher, greybox, helmet, kettle.

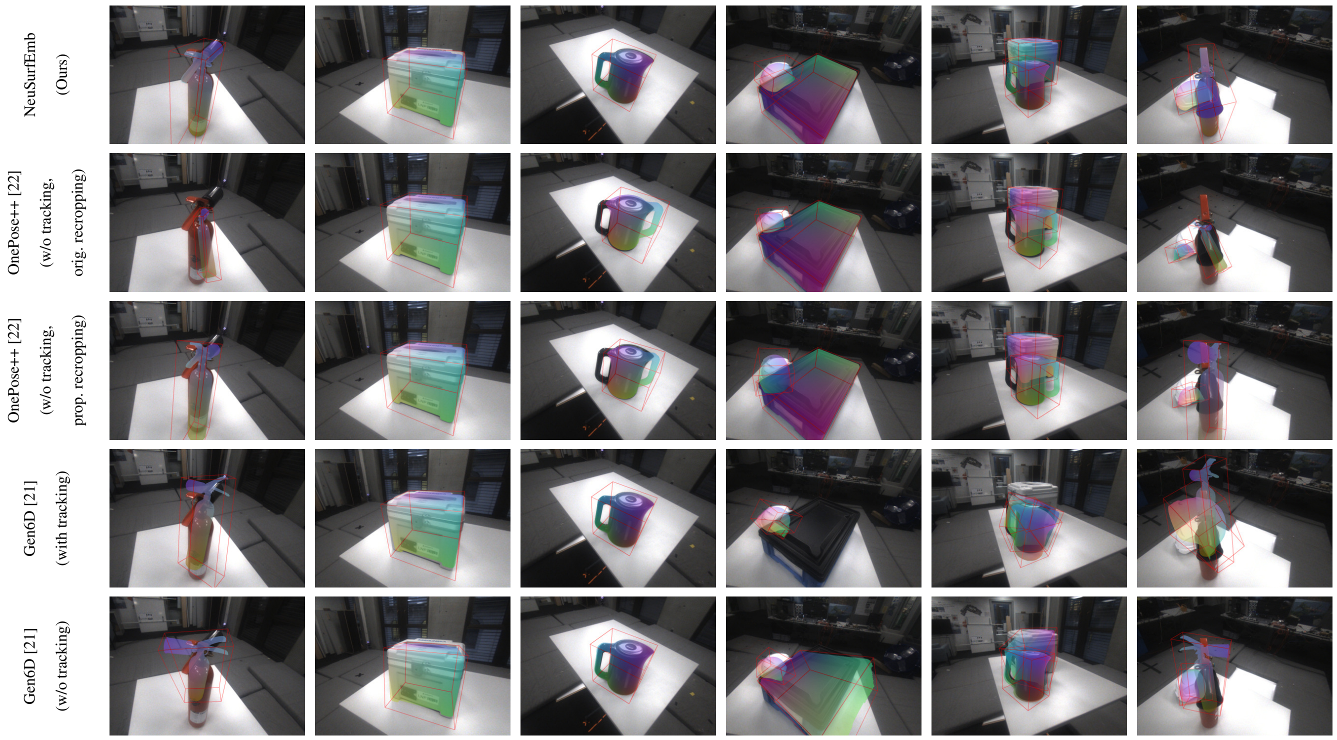
NeuSurfEmb (Ours)
OnePose++ [22] (w/o tracking, orig. recropping)
OnePose++ [22] (w/o tracking, prop. recropping)
Gen6D [21] (with tracking)
Gen6D [21] (w/o tracking)

Fig. 4. Example visualizations of the poses estimated by the different methods in the real-world experiments, displayed as rendered coordinates and reprojected object bounding box overlaid to the original image. The scenes depicted from left to right are: extinguisher, greybox, kettle, bluebox – helmet, greybox – kettle, helmet – extinguisher (cf. Tables IV and V).

are known in the literature to perform worse than CAD-model-based ones [21], [47], and we therefore only reported baselines from the latter category in the dataset evaluations of Sec. IV-C. However, we select [21] and [22] for comparison in our real-world experiments because they represent the best viable option in a robotic scenario. As with our method, both Gen6D and OnePose++ also require a set of views from a "model-training" scene (cf. also Sec. III-A).

For each object, we collect one video recording of the *model-training* scene (Fig. 3), using a regular consumer-grade smartphone, and multiple *evaluation* scenes using a FLIR Firefly S camera. For each object, in one of the evaluation scenes the object is shown in isolation and in the remaining ones an additional object is present in the scene, thereby generating occlusions in several viewpoints (Fig. 4). Note that lighting conditions, background, and color characteristics vary significantly between the model-training and the evaluation scenes, which therefore requires the pose estimation algorithms to be robust to these factors.

To obtain ground-truth camera-to-object poses, needed for evaluation, we track the camera pose in the evaluation scenes through a marker-based motion capture system (Vicon). We then convert the camera-to-marker pose to a camera-to-object pose by defining an object-centered coordinate frame for each object, keeping the position of each object constant across the evaluation recordings, and exploiting the fact that the tracking system coordinate frame stays fixed. Note that the coordinate frame of each object in the model-training scene is in general different from the corresponding one in the evaluation scenes, since the model-training frame is based on the one returned by the SfM pipeline, which is arbitrarily defined. Additionally, given the inherent scale ambiguity of monocular algorithms like SfM, the reference model-training poses, and consequently the output estimated poses, are not expressed in meters, unlike the poses returned by the Vicon system. To register the two coordinate frames and estimate the scale conversion factor, we train a NeuS2 model for both the model-training and the single-object evaluation scene, and we estimate the scaled transform between the two coordinate frames through Iterative Closest Point (ICP) registration

Method	ADD(−S)						5 cm, 5°					
	bluebox⋆	extinguisher	greybox⋆	helmet	kettle	AVG Objects	bluebox	extinguisher	greybox	helmet	kettle	AVG Objects
NeuSurfEmb (Ours)	**1.000**	**0.980**	**1.000**	**0.979**	**0.955**	**0.982**	**0.789**	**0.755**	0.873	**0.937**	**0.803**	**0.825**
OnePose++ [22] (original, with tracking)	0.976	0.515	**1.000**	0.629	0.371	0.683	0.476	0.388	0.853	0.601	0.315	0.510
OnePose++ [22] (w/o tracking, original recropping)	0.994	0.520	**1.000**	0.629	0.315	0.676	0.530	0.388	**0.887**	0.601	0.275	0.519
OnePose++ [22] (w/o tracking, proposed recropping)	**1.000**	0.791	**1.000**	0.650	0.416	0.766	0.530	0.622	0.880	0.587	0.348	0.586
Gen6D [21] (with tracking)	0.795	0.005	**1.000**	0.399	0.663	0.550	0.193	0.000	0.860	0.147	0.281	0.279
Gen6D [21] (w/o tracking)	0.898	0.230	**1.000**	0.294	0.669	0.606	0.042	0.015	0.473	0.112	0.320	0.185

TABLE IV

POSE ESTIMATION PERFORMANCE ON THE REAL-WORLD EXPERIMENTS, NON-OCCLUDED SCENES. ⋆ DENOTES SYMMETRICAL OBJECTS.

Method	AR_{BOP}								5 cm, 5°							
	bluebox − helmet		greybox − kettle		helmet − extinguisher		kettle − bluebox		bluebox − helmet		greybox − kettle		helmet − extinguisher		kettle − bluebox	
	bluebox⋆	helmet	greybox⋆	kettle	helmet	extinguisher	kettle	bluebox⋆	bluebox	helmet	greybox	kettle	helmet	extinguisher	kettle	bluebox
NeuSurfEmb (Ours)	0.937	**0.731**	0.964	**0.752**	**0.918**	**0.869**	**0.587**	0.946	**0.632**	**0.606**	**0.857**	**0.528**	**0.798**	0.664	**0.347**	**0.748**
OnePose++ [22] (original, with tracking)	0.985	0.496	0.982	0.155	0.522	0.587	0.271	0.934	0.588	0.394	0.833	0.056	0.479	0.521	0.068	0.401
OnePose++ [22] (w/o tracking, original recropping)	0.983	0.516	0.994	0.216	0.557	0.524	0.343	**0.991**	**0.632**	0.385	**0.857**	0.153	0.521	0.345	0.102	0.435
OnePose++ [22] (w/o tracking, proposed recropping)	**0.986**	0.444	**0.996**	0.303	0.517	0.776	0.416	0.977	**0.623**	0.269	0.833	0.194	0.445	**0.697**	0.184	0.401
Gen6D [21] (with tracking)	0.002	0.393	0.753	0.237	0.150	0.124	0.056	0.151	0.000	0.087	0.476	0.056	0.059	0.000	0.000	0.000
Gen6D [21] (w/o tracking)	0.558	0.319	0.749	0.548	0.439	0.184	0.473	0.489	0.044	0.058	0.524	0.139	0.084	0.017	0.116	0.122

TABLE V

POSE ESTIMATION PERFORMANCE ON THE REAL-WORLD EXPERIMENTS, OCCLUDED SCENES. ⋆ DENOTES SYMMETRICAL OBJECTS.

between the point clouds of the two NeuS2 models. For both the model-training and the evaluation scenes, we record our videos at a resolution of $1920 \times 1080\,\mathrm{px}$ and sample the recording to obtain approximately 100 frames.

We report the results of our evaluation in Tables IV and V, where we present the performance on each individual object, as well as averaged over all the objects based on their occurrence, for Tab. IV. For the occluded scenes, we only consider an image for evaluation if at least 10% of the object surface is visible. Both baselines implement their own object detector and a tracking module, both of which might introduce additional failures. To ensure a fair evaluation focused uniquely on the pose estimators, we therefore provide ground-truth bounding boxes for each frame to all the methods, but additionally report the performance of the baselines with their original setup. We compute the ground-truth bounding boxes in each evaluation image by rendering object masks using the NeuS2 models and the ground-truth camera poses and fitting a bounding box to the renderings.

Fig. 4 shows qualitative examples of the estimated poses (columns 1-3 for the single-object scenes and column 4-6 for those with occlusions). Both Gen6D and OnePose++ achieve comparable or even slightly better performance than our method on the two symmetrical and geometrically regular objects (bluebox and greybox). However, their performance drops significantly on the remaining objects, which present either more complex geometry or a lower amount of texture. We note in particular that Gen6D achieves low performance on the texture-poor helmet and fails almost completely for the extinguisher (see Tab. IV). OnePose++ achieves slightly better performance on helmet, but fails to make use of the low-texture handle in kettle to discriminate between similar viewpoints, thereby returning poses with rotational errors. Overall, NeuSurfEmb outperforms both Gen6D and OnePose++ by a significant margin across the objects.

Since as observed in the literature [17], the ADD-S metric tends to return large values for symmetric objects and might therefore not be indicative enough of the performance, we additionally report the recall of 5 cm, 5°, thereby not taking symmetries into account. We find that under this metric, NeuSurfEmb achieves better accuracy than the baselines for bluebox, that is, it returns a prediction from the correct side of the object more often than from the opposite one. This indicates that a denser model and explicit image-based training can use texture information to disambiguate object views that appear identical when only considering geometry. Nonetheless, while still achieving close-to-optimal accuracy, our method is slightly outperformed by the baselines for greybox also under the 5 cm, 5° metric; we notice that this performance gap stems mostly from a systematic rotational error that our method produces from specific viewpoints.

A relevant observation, reflected both in the quantitative and qualitative results, is that when enabling tracking, as in their original setup, both baselines generally achieve lower performance. In particular, Gen6D tends to mistakenly track the foreground object in place of the occluded one. An additional element that we notice is that even when providing ground-truth bounding boxes, OnePose++ internally recrops the object detections, which often causes important object features to be ruled out from triangulation, particularly for tall objects such as the extinguisher. We note that simply adapting their code to keep the full object visible when recropping largely increases the performance on these objects, although the accuracy remains lower than that of our method.

Finally, we observe that both Gen6D and OnePose++ have limited ability to handle occlusions, returning significantly inaccurate poses for the occluded object also under relatively mild occlusions (see columns 4 and 6 in Fig. 4). In contrast, NeuSurfEmb shows greater robustness to occlusions, which is also reflected in the better quantitative results.

Overall, we find that our method is able to accurately estimate the object pose across most of the frames when no occlusions are present (98.2% average ADD(−S) over the 5 objects) and shows better robustness and accuracy compared to the available state-of-the-art CAD-model-free baselines, particularly when handling occlusions and objects with limited texture or complex geometry.

V. CONCLUSIONS

We presented a pipeline to train a state-of-the-art 6D object pose estimator from just a small set of input images. We propose forming a NeuS2-based object representation through semi-automated labeling and generating photorealistic training images through NeuS2 rendering with simple cut-and-paste augmentation. These two components obviate the need for both a CAD model and PBR-based image generation, providing a straightforward and practical solution for real-world robotic scenarios. Our method shows competitive performance with respect to state-of-the-art approaches based on CAD models and PBR synthetic data. Finally, our approach outperforms the leading CAD-model-free approaches, demonstrating high accuracy, robustness to mild occlusions, and ease of use in the real world.

References

[1] Y. Wang, Q. Han, M. Habermann, K. Daniilidis, C. Theobalt, and L. Liu, "NeuS2: Fast Learning of Neural Implicit Surfaces for Multi-view Reconstruction," in *ICCV*, 2023. 1, 2, 3

[2] R. L. Haugaard and A. G. Buch, "SurfEmb: Dense and Continuous Correspondence Distributions for Object Pose Estimation with Learnt Surface Embeddings," in *CVPR*, 2022. 1, 2, 3, 4

[3] P. R. Florence, L. Manuelli, and R. Tedrake, "Dense Object Nets: Learning Dense Visual Object Descriptors By and For Robotic Manipulation," in *CoRL*, 2018. 1

[4] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "kPAM: KeyPoint Affordances for Category-Level Robotic Manipulation," in *ISRR*, 2019. 1

[5] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects," in *CoRL*, 2018. 1

[6] N. Hagbi, O. Bergig, J. El-Sana, and M. Billinghurst, "Shape Recognition and Pose Estimation for Mobile Augmented Reality," *IEEE Trans. Vis. Comput. Graph.*, vol. 17, 2010. 1

[7] E. Marchand, H. Uchiyama, and F. Spindler, "Pose Estimation for Augmented Reality: A Hands-On Survey," *IEEE Trans. Vis. Comput. Graph.*, vol. 17, 2015. 1

[8] Y. Su, J. Rambach, N. Minaskan, P. Lesur, A. Pagani, and D. Stricker, "Deep Multi-state Object Pose Estimation for Augmented Reality Assembly," in *IEEE Int. Symp. Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, 2019. 1

[9] M. Rünz and L. Agapito, "Co-Fusion: Real-time Segmentation, Tracking and Fusion of Multiple," in *ICRA*, 2017. 1

[10] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "SLAM++: Simultaneous Localisation and Mapping at the Level of Objects," in *CVPR*, 2013. 1

[11] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "CosyPose: Consistent Multi-view Multi-object 6D Pose Estimation," in *ECCV*, 2020. 1, 4

[12] Z. Li, G. Wang, and X. Ji, "CDPN: Coordinates-Based Disentangled Pose Network for Real-Time RGB-Based 6-DoF Object Pose Estimation," in *ICCV*, 2019. 1, 2, 4

[13] I. Shugurov, F. Li, B. Busam, and S. Ilic, "OSOP: A Multi-Stage One Shot Object Pose Estimation Framework," in *CVPR*, 2022. 1, 2

[14] S. Zakharov, I. Shugurov, and S. Ilic, "DPOD: 6D Pose Object Detector and Refiner," in *ICCV*, 2019. 1, 2

[15] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes," in *ACCV*, 2012. 1, 4

[16] T. Hodaň, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, and X. Zabulis, "T-LESS: An RGB-D Dataset for 6D Pose Estimation of Texture-less Objects," in *WACV*, 2017. 1

[17] T. Hodaň, F. Michel, E. Brachmann, W. Kehl, A. Glent Buch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, F. Manhardt, F. Tombari, T.-K. Kim, J. Matas, and C. Rother, "BOP: Benchmark for 6D Object Pose Estimation," in *ECCV*, 2018. 1, 7

[18] C. Rennie, R. Shome, K. E. Bekris, and A. F. De Souza, "A Dataset for Improved RGBD-Based Object Detection and Pose Estimation for Warehouse Pick-and-Place," *IEEE RA-L*, vol. 1, 2016. 1

[19] T. Hodaň, M. Sundermeyer, B. Drost, Y. Labbé, E. Brachmann, F. Michel, C. Rother, and J. Matas, "BOP Challenge 2020 on 6D Object Localization," in *ECCVW*, 2020. 1, 4

[20] T. Hodaň, V. Vineet, R. Gal, E. Shalev, J. Hanzelka, T. Connell, P. Urbina, S. Sinha, and B. Guenter, "Photorealistic Image Synthesis for Object Instance Detection," in *ICIP*, 2019. 1, 2

[21] Y. Liu, Y. Wen, S. Peng, C. Lin, X. Long, T. Komura, and W. Wang, "Gen6D: Generalizable Model-Free 6-DoF Object Pose Estimation from RGB Images," in *ECCV*, 2022. 1, 2, 5, 6, 7

[22] X. He, J. Sun, Y. Wang, D. Huang, H. Bao, and X. Zhou, "OnePose++: Keypoint-Free One-Shot Object Pose Estimation without CAD Models," in *NeurIPS*, 2022. 1, 2, 5, 6, 7

[23] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment Anything ," *CoRR/2304.02643*, 2023. 1, 2, 3

[24] Y. Cui, C. Jiang, L. Wang, and G. Wu, "MixFormer: End-to-End Tracking with Iterative Mixed Attention," in *CVPR*, 2022. 1, 2, 3

[25] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother, "Learning 6D Object Pose Estimation Using 3D Object Coordinates," in *ECCV*, 2014. 1, 4

[26] M. Denninger, D. Winkelbauer, M. Sundermeyer, W. Boerdijk, M. Knauer, K. H. Strobl, M. Humt, and R. Triebel, "BlenderProc2: A Procedural Pipeline for Photorealistic Rendering," *Journal of Open Source Software*, vol. 8, no. 82, 2023. 2

[27] M. Rad and V. Lepetit, "BB8: A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects without Using Depth," in *ICCV*, 2017. 2

[28] B. Tekin, S. N. Sinha, and P. Fua, "Real-Time Seamless Single Shot 6D Object Pose Prediction," in *CVPR*, 2018. 2

[29] S. Peng, Y. Liu, Q. Huang, H. Bao, and X. Zhou, "PVNet: Pixel-wise Voting Network for 6DoF Pose Estimation," in *CVPR*, 2019. 2, 4

[30] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized Object Coordinate Space for Category-Level 6D Object Pose and Size Estimation," in *CVPR*, 2019. 2

[31] Y. Labbé, L. Manuelli, A. Mousavian, S. Tyree, S. Birchfield, J. Tremblay, J. Carpentier, M. Aubry, D. Fox, and J. Sivic, "MegaPose: 6D Pose Estimation of Novel Objects via Render & Compare," in *CoRL*, 2022. 2

[32] L. Huang, T. Hodaň, L. Ma, L. Zhang, L. Tran, C. Twigg, P.-C. Wu, J. Yuan, C. Keskin, and R. Wang, "Neural Correspondence Field for Object Pose Estimation," in *ECCV*, 2022. 2

[33] B. Wen, J. Tremblay, V. Blukis, S. Tyree, T. Müller, A. Evans, D. Fox, J. Kautz, and S. Birchfield, "BundleSDF: Neural 6-DoF Tracking and 3D Reconstruction of Unknown Objects," in *CVPR*, 2023. 2

[34] H. Chen, F. Manhardt, N. Navab, and B. Busam, "TexPose: Neural Texture Learning for Self-Supervised 6D Object Pose Estimation," in *CVPR*, 2023. 2

[35] D. Dwibedi, I. Misra, and M. Hebert, "Cut, Paste and Learn: Surprisingly Easy Synthesis for Instance Detection," in *ICCV*, 2017. 3

[36] J. L. Schönberger and J.-M. Frahm, "Structure-from-Motion Revisited," in *CVPR*, 2016. 2

[37] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction," in *NeurIPS*, 2020. 2

[38] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *ACM SIGGRAPH Comp. Graph.*, vol. 21, 1987. 2

[39] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, "Implicit Neural Representations with Periodic Activation Functions," in *NeurIPS*, 2020. 3

[40] S. Dombi. (2020) ModernGL, High-performance Python Bindings for OpenGL 3.3+. 3

[41] A. van den Oord, Y. Li, and O. Vinyals, "Representation Learning with Contrastive Predictive Coding," *CoRR/1807.03748*, 2018. 3

[42] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," 2012. 3

[43] M. Afifi and M. Brown, "What Else Can Fool Deep Learning? Addressing Color Constancy Errors on Deep Neural Network Performance," in *ICCV*, 2019. 3

[44] T. Hodaň, J. Matas, and Š. Obdržálek, "On Evaluation of 6D Object Pose Estimation," in *ECCVW*, 2016. 4

[45] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images," in *CVPR*, 2013. 4

[46] T. Hodaň, D. Baráth, and J. Matas, "EPOS: Estimating 6D Pose of Objects with Symmetries," in *CVPR*, 2020. 4

[47] J. Sun, Z. Wang, S. Zhang, X. He, H. Zhao, G. Zhang, and X. Zhou, "OnePose: One-Shot Object Pose Estimation without CAD Models," in *CVPR*, 2022. 6