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.

In-Hand 3D Object Scanning from an RGB Sequence

Shreyas Hampali[1,3*] Tomas Hodan[1] Luan Tran[1]
Lingni Ma[1] Cem Keskin[1] Vincent Lepetit[2,3]

[1]Reality Labs at Meta
[2]LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France
[3]Institute for Computer Graphics and Vision, Graz University of Technology, Graz, Austria

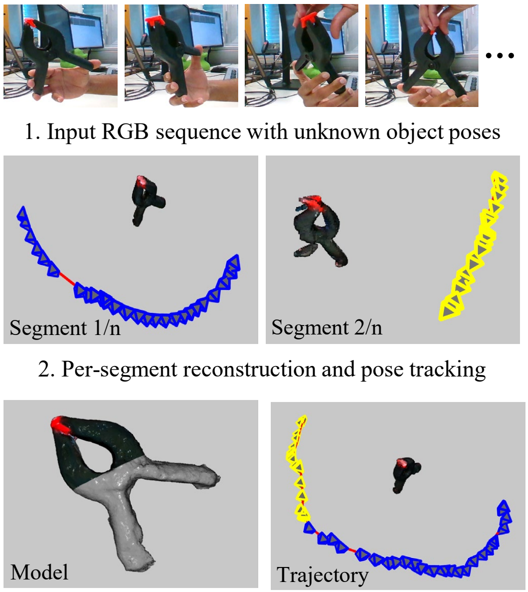
Project Website: https://rgbinhandscanning.github.io/

Abstract

We propose a method for in-hand 3D scanning of an unknown object with a monocular camera. Our method relies on a neural implicit surface representation that captures both the geometry and the appearance of the object, however, by contrast with most NeRF-based methods, we do not assume that the camera-object relative poses are known. Instead, we simultaneously optimize both the object shape and the pose trajectory. As direct optimization over all shape and pose parameters is prone to fail without coarse-level initialization, we propose an incremental approach that starts by splitting the sequence into carefully selected overlapping segments within which the optimization is likely to succeed. We reconstruct the object shape and track its poses independently within each segment, then merge all the segments before performing a global optimization. We show that our method is able to reconstruct the shape and color of both textured and challenging texture-less objects, outperforms classical methods that rely only on appearance features, and that its performance is close to recent methods that assume known camera poses.

1. Introduction

Reconstructing 3D models of unknown objects from multi-view images is a computer vision problem which has received considerable attention [8]. With a single camera, a user can capture multiple views of an object by manually moving the camera around a static object [22, 26, 43] or by turning the object in front of the camera [27, 31, 35, 36]. The latter approach is often referred to as *in-hand object scanning* and is convenient for reconstructing objects from cameras mounted on an AR/VR headset such as Microsoft HoloLens or Meta Quest. Moreover, this approach can reconstruct the full object surface, including the bottom part which cannot be scanned in the static-object setup.



1. Input RGB sequence with unknown object poses

2. Per-segment reconstruction and pose tracking

3. Object model and pose trajectory after global optimization

Figure 1. Given an RGB sequence of a hand manipulating an unknown object, our method reconstructs the 3D shape and color of the object, even if the object surface is non-Lambertian or poorly textured. We first split the input sequence into multiple overlapping segments (two in this figure) in which the object can be reliably reconstructed and tracked. We then use the tracked object-camera relative poses to initialize a global optimization that produces the final object model and refined pose trajectory.

Recent 3D reconstruction methods rely on neural representations [19, 21–23, 42, 43]. By contrast with earlier reconstruction methods [9], the recent methods can provide an accurate dense 3D reconstruction even in non-Lambertian conditions and without any prior knowledge of the object shape. However, most of these methods assume that the camera poses are provided, typically by Structure-from-Motion (SfM) methods such as COLMAP [29]. Apply-

[*] Work done as part of Shreyas's PhD thesis at TU Graz, Austria.

ing SfM methods to in-hand object scanning is problematic as these methods require a sufficient number of distinct visual features and can thus handle well only textured objects. NeRF-based methods such as [13, 15, 16, 34], which simultaneously estimate the radiance field of the object and the camera poses without requiring initialization from COLMAP, are restricted to forward-facing camera captures. As we experimentally demonstrate, these methods fail to converge if the images cover a larger range of viewpoints, which is typical for in-hand scanning.

We propose a method for in-hand object scanning from an RGB image sequence with unknown camera-object relative poses. We rely on a neural representation that captures both the geometry and the appearance of the object and therefore enables reconstructing even poorly textured objects, as shown in Fig. 1. By contrast with most NeRF-based methods, we do not assume that the camera poses are available and instead simultaneously optimize both the object model and the camera trajectory. As global optimization over all input frames is prone to fail, we propose an incremental optimization approach. We start by splitting the sequence into carefully selected overlapping segments within which the optimization is likely to succeed. We then optimize our objective for incremental object reconstruction and pose tracking within each segment independently. The segments are then combined by aligning poses estimated at the overlapping frames, and we finally optimize the objective globally over all frames of the input sequence to achieve complete object reconstruction.

Note that we do not make any assumptions on the type of hand-object grasps and also consider scenarios where the grasping is dynamic, i.e., contact points continuously change, which corresponds to natural hand-object interactions. This is in contrast with the recent work [11] that considers only static grasps. In fact, we refrain from considering hand poses in our method as they cannot be reliably estimated under occlusion in case of dynamic grasps, which could lead to incorrect object reconstruction.

We experimentally demonstrate that the proposed method is able to reconstruct the shape and color of both textured and challenging texture-less objects. We evaluate the method on datasets HO-3D [7], RGBD-Obj [32] and on the newly captured sequences with challenging texture-less objects. We show that the proposed method achieves higher-quality reconstruction than COLMAP [29], which fails to estimate the object poses in the case of poorly textured objects and is in par with a strong baseline method which uses ground-truth object poses. Our method also outperforms a very recent single-image based object reconstruction method [44], even though this method is trained on sequences of the same object. We also demonstrate the real-world applicability of our method by in-hand scanning an object with ARIA glasses [18] (see supplement).

2. Related Work

This section reviews previous methods for in-hand scanning and general object reconstruction from color images, and compares them with the method proposed in this paper.

2.1. In-Hand Object Scanning

Using an RGB-D sensor, several in-hand scanning systems [27,33,35–37] rely on tracking and are able to recover the shape of small objects manipulated by hands. Later, [31] showed how to use the motion of the hand and its contact points with the object to add constraints useful to deal with texture-less and highly symmetric objects, while restricting the contact points to stay fixed during the scanning. Unfortunately, the requirement for an RGB-D sensor limits applications of these techniques.

More recently, with the development of deep learning, several methods have shown that it is possible to infer the object shape from a single image [10, 14, 44] after training on images of hands manipulating an object with annotations of the object pose and shape. Given the fact that the geometry is estimated from a single image, the results are impressive. However, the reconstruction quality is still limited, especially because these methods do not see the back of the object and cannot provide a good prediction of the appearance of the object for all possible viewpoints. In this paper, we propose an approach for in-hand object scanning which estimates the shape and color of a completely unknown object from a sequence of RGB images, without any pre-training on annotated images.

2.2. Reconstruction from Color Images

Recovering the 3D geometry of a static scene and the camera poses from multiple RGB images has a long history in computer vision [5, 6, 9, 30]. Structure-from-Motion (SfM) methods are now very robust and accurate, however they are limited to scenes with textures, which is not the case for many common objects.

In the past few years, with the emergence of neural implicit representations as effective means of modeling 3D geometry and appearance, many methods [20–22, 42, 43] reconstruct a 3D scene by optimizing a neural implicit representation from multi-view images by minimising the discrepancy between the observed and rendered images. These methods achieve impressive reconstructions on many scenes, but they still need near perfect camera poses, which are typically estimated by Structure-from-Motion methods.

Several NeRF-based methods have attempted to retrieve the camera poses while reconstructing the scene. Methods such as NeRF-- [34], SCNeRF [13] and BARF [16] show that camera poses can be estimated even when initialized with identity matrix or random poses while simultaneously estimating the radiance field. However, these methods are shown to converge only on forward facing scenes and require coarse initialization of poses for 360° captures as in in-hand object scanning. More recently, SAMURAI [2]) used manual rough quadrant annotations for coarse-level pose initialization and showed that object shape and material can be recovered along with the accurate camera poses.

In this work we propose to estimate the camera-object relative pose in from RGB image sequence and reconstruct the object shape without any prior information of the object or its poses. Unlike previous methods, we rely on the temporal information and incrementally reconstruct the object shape and estimate its pose.

3. Proposed Method

In this section, we first describe the considered setup and how we represent the object with a neural representation. Then, we derive an objective function for estimating the object reconstruction and the camera pose trajectory, and explain how we optimize this function.

3.1. In-Hand Object Scanning Setup

Input and Output. Our input is a sequence of RGB images showing an unknown rigid object being manipulated by one or two hands in the field of view of the camera. The output is a color 3D model of the manipulated object. The input sequence is captured by an egocentric camera or a camera mounted on a tripod. In both cases, the relative pose between the camera and the object is unknown. In order to achieve full reconstruction of the object, the image sequence is assumed to show the object from all sides.

Available Information About Objects and Hands. The segmentation masks of the object and hands are assumed available for all input images. In our experiments, we obtain the masks by of-the-shelf networks – by Detic [45], which can segment unknown objects in a single RGB image, or by DistinctNet [1] which can segment an unknown moving object from a pair of images with static background. We additionally use segmentation masks from SeqFormer [38] to ignore pixels of hands that manipulate the object.

Phong Reflection Model and Distant Lights. The object to reconstruct is assumed to be solid (*i.e.*, non-translucent), and we model the reflectance properties of the object surface with the Phong reflection model [12] and assume that the light sources are far from the object and the camera. Under the Phong model, the observed color at a surface point depends on the viewing direction, the surface normal direction, and the light direction. If the light sources are far, the incoming light direction can be approximated to remain unchanged, which allows us to use the standard neural radiance field [20] to model the object appearance. This assumption is reasonable as rotation is the primary transformation of the object during in-hand manipulation–on the HO-3D dataset, which contains sequences of a hand manipulating objects, the maximum standard deviation of the object's 3D location is only 7.9cm.

3.2. Object Representation

Implicit Neural Fields. As in UNISURF [22], we represent the object geometry by an occupancy field and the object appearance by a color field, with each realized by a neural network. The occupancy field is defined as a mapping: $o_\theta(\mathbf{x}) : \mathbb{R}^3 \rightarrow [0, 1]$, where θ represents the parameters of the network and $\mathbf{x}$ is a 3D point in the object coordinate system. The object surface is represented by 3D points $\mathcal{S} = \{\mathbf{x} \,|\, o_\theta(\mathbf{x}) = 0.5\}$, and the surface mesh can be recovered by the Marching Cubes algorithm [17].

The color field is a mapping: $c_\theta(\mathbf{x}; \mathbf{d}, \mathbf{n}, \mathbf{h}) : \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^n \rightarrow \mathbb{R}^3$ that represents the color at a surface point $\mathbf{x} \in \mathcal{S}$ and is conditioned on the viewing direction $\mathbf{d}$ (*i.e.*, the direction from the camera center to the point $\mathbf{x}$), the normal vector $\mathbf{n}$ at $\mathbf{x}$, and the geometry feature $\mathbf{h}$ at $\mathbf{x}$ which has n dimensions and is extracted from the occupancy field network. The color for a particular pixel/ray $\mathbf{r}$ is defined as $\hat{C}_i(\mathbf{r}) = c_\theta(\mathbf{x}_s)$, where $\mathbf{x}_s$ is the closest point on the object surface along ray $\mathbf{r}$ (the object is assumed non-translucent). To simplify the notation, we include in θ both the parameters of the occupancy field and of the color field as these two networks are optimized together.

Rendering. As in [22], the rendered color at a pixel in a frame i is obtained by integrating colors along the ray $\mathbf{r}$ originating from the camera center and passing through the pixel. The continuous integration is approximated as:

$$\hat{C}_i(\mathbf{r}) = \sum_{k=1}^{M} \gamma(\mathbf{x}_k) c_\theta(\mathbf{x}_k; \mathbf{d}_k, \mathbf{h}_k, \mathbf{n}_k) \tag{1}$$

$$\text{with } \gamma(\mathbf{x}_k) = o_\theta(\mathbf{x}_k) \prod_{l<k} \left(1 - o_\theta(\mathbf{x}_l)\right) , \tag{2}$$

where $\{\mathbf{x}_k\}$ are M samples along the ray $\mathbf{r}$. The alpha-blending coefficient $\gamma(\mathbf{x}_k)$, is defined as in [22], is 1 if point $\mathbf{x}_k$ is on the visible surface of the object and 0 otherwise.

3.3. Reconstruction Objective

In UNISURF [22], the network parameters θ are estimated by solving the following optimization problem:

$$\theta^* = \arg\min_\theta \sum_i \sum_{\mathbf{r} \in \mathcal{R}_i} \mathcal{L}^i_{\text{c}}(\mathbf{r}) , \tag{3}$$

$$\mathcal{L}^i_{\text{c}}(\mathbf{r}) = ||\hat{C}_i(\mathbf{r}) - C_i(\mathbf{r})|| , \tag{4}$$

where $\mathcal{L}^i_{\text{c}}(\mathbf{r})$ is the photometric loss measuring the difference between the rendered color $\hat{C}_i(\mathbf{r})$ and the observed color $C_i(\mathbf{r})$ at a pixel intersected by the ray $\mathbf{r}$ in the frame i, and $\mathcal{R}_i$ is the set of rays sampled in the frame i.

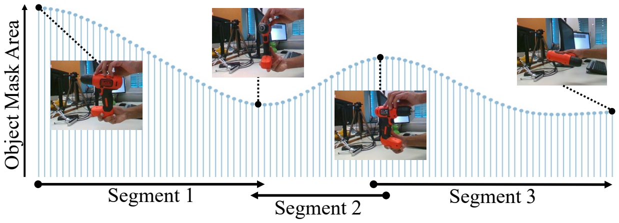


Figure 2. **Splitting a sequence into easy-to-track segments.** The segment boundaries are defined at frames with locally maximal or minimal area of the object mask (the start and the end of each segment is shifted by a few frames from the extremum to make the segments overlap). Note that we can track backwards in time, from a local maximum to a local minimum.

In our case, we additionally optimize the camera poses:

$$\theta^*, \{\mathcal{T}_i^*\} = \underset{\theta,\{\mathcal{T}_i\}}{\arg\min} \sum_i \Big(\sum_{\mathbf{r}\in\mathcal{H}_i} \mathcal{L}_\mathrm{c}^i(\mathbf{r}) + \sum_{\mathbf{r}\in\mathcal{M}_i} \mathcal{L}_\mathrm{s}^i(\mathbf{r}) \Big) , \quad (5)$$

where $\mathcal{T}_i$ is the camera pose of frame i expressed by a rigid transformation from the camera coordinate system to the object coordinate system. $\mathcal{H}_i$ is the set of object rays in frame i, and $\mathcal{M}_i$ is the set of object and background rays in frame i. We only use rays passing through the object and background pixels and ignore the hand pixels. The term $\mathcal{L}_\mathrm{s}^i(\mathbf{r})$ is a segmentation loss for ray $\mathbf{r} \in \mathcal{M}_i$:

$$\mathcal{L}_\mathrm{s}^i(\mathbf{r}) = \mathrm{BCE}\Big(\max_k \ \{o_\theta(\mathbf{x}_k)\}, S_i(\mathbf{r}) \Big) , \quad (6)$$

where BCE$(\cdot)$ is the binary cross-entropy loss, and $S_i(\mathbf{r})$ is the object mask value for ray $\mathbf{r}$ in the frame i (the mask is obtained as described in Sec. 3.1). The value of $S_i(\mathbf{r})$ is 1 if the pixel corresponding to ray $\mathbf{r}$ lies in the provided object mask, and 0 otherwise. The term $\max_k \ \{o_\theta(\mathbf{x}_k)\}$ is the maximum occupancy along the ray $\mathbf{r}$ according to the estimated occupancy field $o_\theta(.)$, and is expected to be 1 if $\mathbf{r}$ intersects the object and 0 otherwise.

3.4. Optimization

Directly optimizing Eq. (5) is prone to fail. As we show in Sec. 5, a random (or a fixed) initialization of poses followed by an optimization procedure similar to the one used in BARF [16] leads to degenerate solutions. Instead, we propose an incremental optimization approach which starts by splitting the sequence into carefully selected overlapping segments, within which the optimization is more likely to succeed (Sec. 3.4.1). We optimize the objective in each segment by incremental frame-by-frame reconstruction and tracking, with the objective being extended by additional loss terms to stabilize the tracking (Sec. 3.4.2). Then, we merge the segments by aligning poses estimated at the overlapping frames (Sec. 3.4.3), and finally optimize the objective globally over all frames of the sequence (Sec. 3.4.3).

3.4.1 Input Sequence Segmentation

We observed in our early experiments that frame-to-frame tracking is prone to fail when previously observed parts of the object start disappearing and new parts start appearing. This is not surprising as there is no 3D knowledge about the new parts yet, and the current reconstruction of the object is disappearing and cannot be used to track these new parts.

We therefore propose to split the sequence into segments so that tracking on each segment is unlikely to drift much. How can we detect when new parts are appearing? We observe that this can be done based on the apparent area[1] of the object: Under the assumption that the distance of the object to the camera and the occlusions by the hand do not change much, large parts of the object disappear when the apparent area goes through a minimum (see Figure 2). We therefore split the input sequence into multiple segments, with their boundaries defined at frames where the apparent area reaches local maxima and minima, and process each segment from the local maximum to the local minimum (*e.g.*, segments 1 and 3 in Figure 2 are processed from left to right and segment 2 is processed from right to left). The local extrema are computed from the smoothed curve of the apparent object area using a sliding window with the length of 12 frames. The start and the end of each segment is shifted by a few frames from the extremum to introduce overlaps with the neighboring segments (the overlaps are used in Sec. 3.4.3 to merge the estimated per-segment pose trajectories). With this approach, tracking within a segment starts with a point of view where the object reprojection area is large in the image, which facilitates bootstrapping of the tracking together with our shape regularization loss.

3.4.2 Per-Segment Optimization

Within each segment, we iteratively optimize the following objective on a progressively larger portion of the segment allowing us to incrementally reconstruct the object and track its pose. The T frames in a segment are denoted by the set $\{\mathcal{S}\}_{i=1}^T$. Over the course of the optimization, the index t of the currently considered frame progresses from the first to the last frame of the segment, and for each step we solve:

$$\theta^*, \{\mathcal{T}_i^*\}_{i=1}^t = \underset{\theta,\{\mathcal{T}_i\}_{i=1}^t}{\arg\min} \quad \sum_{i=1}^t \sum_{\mathbf{r}\in\mathcal{H}_i} \mathcal{L}_\mathrm{c}^i(\mathbf{r}) + ... \quad (7)$$

$$\sum_{i=1}^t \sum_{\mathbf{r}\in\mathcal{M}_i} \Big(\mathcal{L}_\mathrm{s}^i(\mathbf{r}) + \mathcal{L}_\mathrm{f}^i(\mathbf{r}) + \mathcal{L}_\mathrm{r}^i(\mathbf{r}) \Big) + \sum_{i=1}^{t-1} \sum_{\mathbf{r}\in\mathcal{M}_i} \mathcal{L}_\mathrm{d}^i(\mathbf{r}) \, .$$

The terms $\mathcal{L}_\mathrm{c}^i$ and $\mathcal{L}_\mathrm{s}^i$ are the color and mask losses defined in Eq. (5), $\mathcal{L}_\mathrm{f}^i$ is a loss based on optical flow that

[1] As mentioned earlier, we obtain a mask of the object by segmenting the image, so we can easily compute its apparent area.

provides constraints on the poses, $\mathcal{L}_{\text{r}}^{i}$ is a shape regularization term that prevents degenerate object shapes, and $\mathcal{L}_{\text{d}}^{i}$ is a synthetic-depth loss that stabilizes the tracking. $\mathcal{H}_i$ is the set of rays going through pixels on the object in frame i, and $\mathcal{M}_i$ the set of rays going through pixels on the object or the background in frame i. More details on the loss terms $\mathcal{L}_{\text{f}}^{i}$, $\mathcal{L}_{\text{r}}^{i}$ and $\mathcal{L}_{\text{d}}^{i}$ are provided later in this section.

The network parameters θ are initialized by their estimate from the previous iteration $t-1$. The camera pose $\mathcal{T}_t$ for $t > 1$ is initialized using the second-order motion model applied to the previous poses $\{\mathcal{T}_i\}_{i=0}^{t-1}$. The first camera pose $\mathcal{T}_0$ is initialized to a fixed distance from the origin of the object coordinate system and orientated such that the image plane faces the origin. At each iteration, we sample a fixed percentage of rays from the new frame (set to 15% empirically) and the rest from the previous frames.

Optical Flow Loss. This term provides additional constraints on the camera poses and is defined as:

$$\mathcal{L}_{\text{f}}^{i}(\mathbf{r}) = \sum_k \gamma(\mathbf{x}_k)\big(\pi_i(\mathbf{x}_k) - \pi_{i\text{-}1}(\mathbf{x}_k) - \text{OF}_i(\pi_{i\text{-}1}(\mathbf{x}_k))\big)^2 , \qquad (8)$$

where $\{\mathbf{x}_k\}_i$ are 3D points along ray $\mathbf{r}$, $\pi_i(\mathbf{x})$ is the 2D reprojection of the point $\mathbf{x}$ in the frame i, OF_i is the optical flow between frame $i-1$ and frame i, and $\gamma(\cdot)$ is as defined in Eq. (2) and evaluates to one for points on the object surface and zero elsewhere. Fig. 3 shows the effect of optical flow loss on the trajectory after several optimization steps. We use [40] to compute the optical flow.

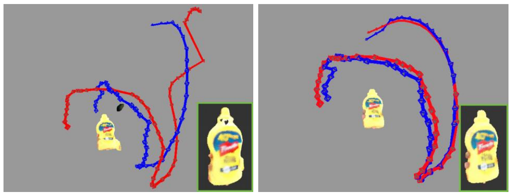


Figure 3. **Effect of the optical flow loss $\mathcal{L}_{\text{f}}$.** Pose estimates (red) are more stable when the loss $\mathcal{L}_{\text{f}}$ is applied (right). The ground-truth poses are shown in blue.

Shape Regularization Loss. During early iterations (*i.e.*, when t is small), the occupancy field is under-constrained and needs to be regularized to avoid degenerate object shapes. We introduce a regularization that encourages reconstruction near the origin of the object coordinate system:

$$\mathcal{L}_{\text{r}}^{i}(\mathbf{r}) = \sum_k o_\theta(\mathbf{x}_k)\exp\left(\alpha\cdot\|\mathbf{x}_k\|_2\right) , \qquad (9)$$

where α is a hyperparameter. At $t = 0$, minimizing $\mathcal{L}_{\text{r}}^{i}$ results in an object surface that is parallel to the image plane (see supplement for explanation). Encouraging a planar proxy as an approximation of the object shape helps to stabilize the early stage of the optimization. Fig. 4 shows examples achieved with and without the regularization.

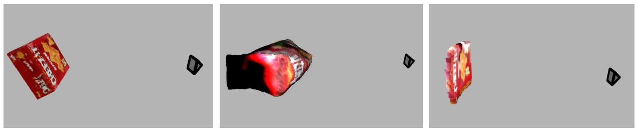


Figure 4. **Effect of the shape regularization loss $\mathcal{L}_{\text{r}}$.** Left to right: The ground-truth object mesh, implicit surface reconstructed without the regularization term at $t = 1$, and implicit surface reconstructed with the regularization term at $t = 1$.

Synthetic-Depth Loss. We also introduce a loss based on synthetic depth maps rendered by the object shape estimate. The motivation for this term is to regularize the evolution of the shape estimate and prevent its drift. It is defined as:

$$\mathcal{L}_{\text{d}}^{i}(\mathbf{r}) = \Big(\sum_k \gamma(\mathbf{x}_k)\text{dep}_i(\mathbf{x}_k) - \hat{d}_i(\mathbf{r})\Big)^2 , \qquad (10)$$

where $\text{dep}_i(\mathbf{x}_k)$ is the depth of the point $\mathbf{x}_k$ along the ray $\mathbf{r}$, $\gamma(\cdot)$ is as defined in Eq. 2 and $\hat{d}_i$ is the depth map rendered using the previous estimates of the object model and the camera pose for frame i. Note that $\mathcal{L}_{\text{d}}^{i}$ is only applied on rays from frames $[1, t-1]$ at optimization step t of Eq. (7) whose synthetic depths are pre-computed. Figure 5 illustrates the contribution of this term.

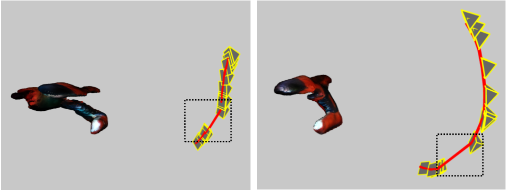


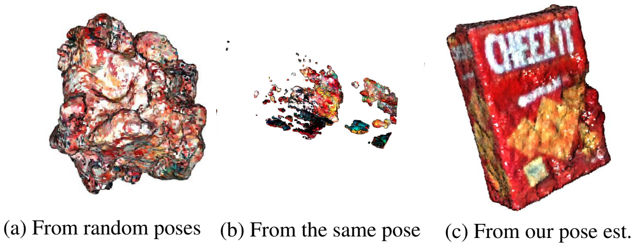
Figure 5. **Effect of the synthetic-depth loss $\mathcal{L}_{\text{d}}$.** Large pose changes (highlighted by black boxes) can deform previously reconstructed parts of the object if the depth loss $\mathcal{L}_{\text{d}}$ is not used (left).

3.4.3 Global Optimization

The camera trajectories and object reconstruction for each segment are recovered up to a rigid motion and a scaling factor. To express the overlapping segments in a common coordinate frame, we align the pose estimates at the overlapping frames with the following procedure. Let $\mathcal{T}_i^k = [\phi_i, t_i]$ be the rotation and translation of the camera for the frame i in the segment k with N_s frames. We obtain a normalized pose by taking $\hat{\mathcal{T}}_i^k = \left[\phi_i; t_i/\frac{1}{N_s}\sum_j \|t_j\|\right]$. We then retrieve the rigid motion $\mathcal{T}_{k_1\to k_2}$ (rotation and translation) that aligns two overlapping segments k_1 and k_2:

$$\mathcal{T}_{k_1\to k_2} = \arg\min_{\mathcal{T}} \sum_i \|\mathcal{T}\cdot\hat{\mathcal{T}}_i^{k_1} - \hat{\mathcal{T}}_{\mathcal{N}(i)}^{k_2}\|_F , \qquad (11)$$

where $\|\cdot\|_F$ denotes the Frobenius norm, $\mathcal{N}(i)$ is the frame index in segment k_2 corresponding to frame i in segment k_1, and the summation is over the set of all overlapping frames. In practice, we observed that as less as a single overlapping frame is sufficient for connecting the segments.



(a) From random poses (b) From the same pose (c) From our pose est.

Figure 6. **Reconstruction from different initial poses.** Only initialization from coarse pose estimates yield a meaningful solution.

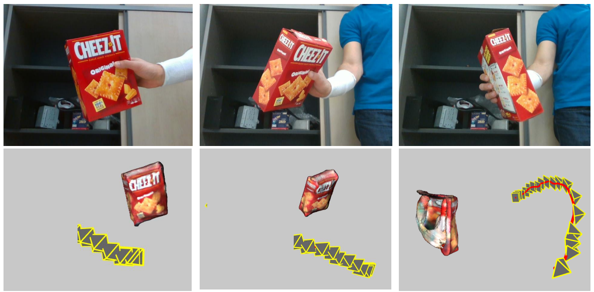


Figure 7. **Incremental reconstruction and pose tracking is prone to fail beyond the segment boundary.** On this representative example, the incremental reconstruction and pose tracking procedure works well as long as the front face is visible. When the front face starts to disappear and new parts start to appear (last column), reconstruction degrades and pose tracking drifts.

We use the aligned poses of two neighboring segments as pose initialization and optimize the objective function from Eq. (5). The network parameters θ are initialized to reconstruct a sphere. The neighboring segments are combined iteratively until we obtain complete reconstruction from the full sequence. Fig. 6 shows the reconstruction with different pose initializations – even for a textured object, coarse initialization is necessary for convergence. In Fig. 7, we show a situation where the incremental reconstruction and pose tracking continues beyond the segment boundary – the solution degrades when new surface parts appear.

4. Implementation Details

The occupancy and color field networks are implemented by 8-layer MLP's with ReLU activations and a hidden dimension of F. Fourier features [20] at k_x octaves are used to encode the 3D coordinates, and at k_d octaves to encode the view direction. During the per-segment optimization, similar to [41], instead of directly optimizing the 6D pose parameters, the pose is parameterized with a CNN that takes the RGB image as input and outputs the 6DoF pose. Weights of the CNN are initialized with weights pre-trained on ImageNet [4]. The CNN provides a neural basis for the pose parameters and acts as a regularizer. Without the CNN parameterization, the per-segment optimization procedure described in the section Sec. 3.4.2 typically fails.

During the per-segment optimization (Sec. 3.4.2), we set $F = 128$, $k_x = 4$, $k_d = 2$ and run 6k gradient descent iterations at each tracking step. Further, at each step, we add 5 frames to increase the optimization speed.

For the global optimization (Sec. 3.4.3), we use $F = 256$, $k_x = 8$, $k_d = 4$ and run 25k gradient descent iterations for a pair of segments. We use smooth masking of frequency bands as described in BARF [16] for better convergence and optimize the 6D pose variables directly instead of using CNN parameterization in this stage. The frames are subsampled such that their maximum number is 150.

We compute the local maxima and minima from the object area curve as explained in Sec. 3.4.1 by first performing a Gaussian filtering of per frame object areas.

5. Experiments

We evaluate our method quantitatively and qualitatively on the HO-3D dataset [7], which contains sequences of objects from the YCB dataset [39] being manipulated by one hand. We also show qualitative results on the RGB images from the RGBD-Obj dataset [32] and on sequences that we captured for this project and that show two challenging texture-less YCB objects: the clamp and the cube. The latter two datasets feature two hands but do not provide object pose annotations. We evaluate the accuracy of the reconstructed shape and color, and of the estimated poses.

3D Reconstruction Metric. As in [24], we first align the estimated object mesh with the ground-truth mesh by ICP and then calculate the RMSE Hausdorff distance from the estimated mesh to the ground truth mesh. As our meshes are only estimated up to a scaling factor, we allow the meshes to scale during the ICP alignment.

Object Texture Metrics. The recovered object texture is compared with the ground truth using the PSNR, LPIPS and SSIM metrics. Specifically, we render the recovered object appearance from the ground truth poses for images that were not used in the optimization and compare the renderings with the images. Since the pose has to be accurate to obtain reliable metrics, we first perform photometric optimization on the trained model to obtain accurate poses and then render the images as in BARF [16].

Pose Trajectory Metric. As the 3D model and poses are recovered up to a 3D similarity transformation, we first align the estimated poses with the ground truth by estimating the transformation between the two. We then calculate the absolute trajectory error (ATE) [3, 13, 25] and plot the percentage of frames for which the error is less than a threshold. We use the area under curve of the ATE plot as the metric.

5.1. Evaluation on HO-3D

HO-3D contains 27 multi-view sequences (68 single-view sequences) of hand-object interactions with 10 YCB objects [39] annotated with 3D poses. We consider the same

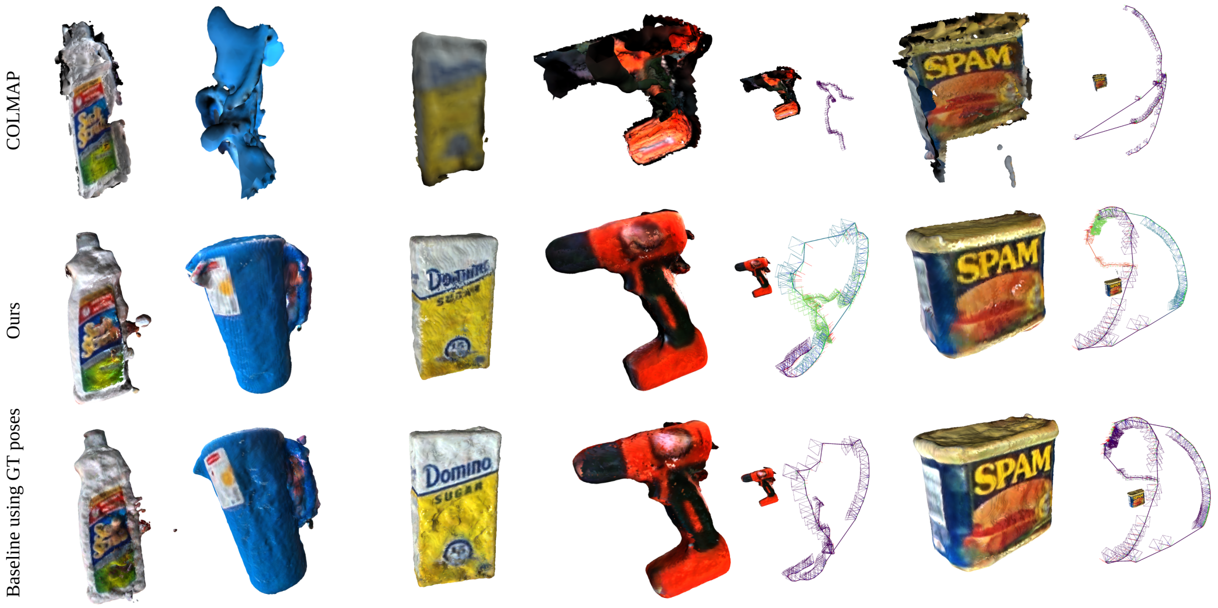


Figure 8. **Reconstructed models and pose trajectories on HO-3D** for COLMAP [28] (top row), our method (middle row), and the UNISURF [22] baseline that uses ground-truth poses (bottom row). We restrict the keypoint matches in COLMAP to only object pixels using the segmentation masks obtained from a pre-trained network (Sec. 3.1). COLMAP recovers only incomplete pose trajectories in absence of texture, which leads to incomplete or failed reconstructions. Our method relies on both geometric and texture features and produces reliable pose estimates, which results in similar reconstructions as produced by the strong baseline relying on ground-truth poses.

multi-view sequences as in [24] for the 3D reconstruction. As the ground-truth 3D poses are provided in this dataset, we also evaluate the accuracy of our estimated poses along with the reconstruction and texture accuracy.

Baselines. We compare with COLMAP [29], the single-image object reconstruction method by Ye *et al*. [44], the RGB-D reconstruction method by Patten *et al*. [24], and UNISURF [22]. The last two methods rely on the ground-truth camera poses, whereas the other methods (including ours) do not. In the case of [24], we compare only the 3D reconstruction accuracy with this method as the pose and object texture evaluations are not reported. We obtain results from COLMAP using the sequential keypoint matching technique, and set the mesh trim parameter to 10. The largest connected component in the reconstructed mesh is then selected as the final result. We observed that COLMAP fails to obtain complete reconstruction for most objects due to insufficient keypoint matches and results in multiple non-overlapping partial reconstructions, which cannot be combined. The method by Ye *et al*. [44] uses a single input image but is pre-trained on sequences of the same object. Note that our method is not pre-trained and thus the reconstructed object is completely unknown to the method.

Results. Table 1 compares the one-way RMSE Hausdorff distance of our method with COLMAP, the single-frame method of [44], and the RGBD method of [24]. We calculate the average metric over the sequence for [44]. Our method consistently achieves higher performance than COLMAP on all objects and higher performance over [44] on average. Our method is competitive with the RGBD-based method and the strong baseline for most objects. COLMAP fails to obtain keypoint matches on banana and scissors. The lower accuracy of our method on pitcher and banana is due to the lack of both geometric and texture features. COLMAP achieves accurate pose trajectories only for the cracker box and sugar box as they contain rich image features on all the surfaces. The lower accuracy of COLMAP on other objects is due to poor texture (Figure 8).

Table 2 shows the area under curve (AUC) of the absolute trajectory error (ATE) plot with the maximum ATE threshold of 10 cm. Our method outperforms COLMAP, which cannot recover the complete trajectory for many objects. Both our method and COLMAP fail to obtain meaningful reconstruction for scissors due to its thin structure.

In Table 3, we provide the PSNR, SSIM and LPIPS metrics for our proposed method and the strong baseline that uses the ground-truth poses. Our method achieves similar accuracy as the baseline method on all objects, despite the fact that our method estimates the poses instead of using the ground-truth ones. Results of our method, the UNISURF baseline and COLMAP are shown in Figure 8.

5.2. Evaluation on RGBD-Obj and New Sequences

Qualitative results from an RGBD-Obj [32] sequence showing the mustard bottle and from two new sequences

Object	Ye *et al.* [44]	COLMAP [28]	Ours	UNISU. [22]	RGBD [24]
3: cracker box	10.21	4.08	**2.91**	3.40	3.54
4: sugar box	6.19	6.66	**3.01**	3.49	3.34
6: mustard	2.61	**4.43**	4.44	4.34	3.28
10: potted meat	3.43	10.21	**1.95**	1.54	3.26
21: bleach	**4.18**	14.11	5.63	3.41	2.43
35: power drill	15.15	11.06	**5.48**	5.33	3.77
19: pitcher base	**8.87**	43.38	9.21	4.63	4.73
11: banana	**3.47**	-	4.60	3.98	2.44
Average	6.76	13.41	**4.65**	3.76	3.34

Table 1. **RMSE Hausdorff distance (mm) from the estimated to the ground-truth 3D model.** UNISURF [22] and RGBD [24] are strong baselines as they use ground-truth poses, and the latter also depth images. Our object reconstructions are close to the baselines, even though we do not use the ground-truth poses, and systematically better than COLMAP, which slipped on the banana.

Object	3	4	6	10	21	35	19	25	11	Avg
COLMAP	7.4	**7.4**	3.5	0.1	1.5	2.8	4.1	**2.4**	0.0	2.9
Ours	**7.6**	6.8	**5.2**	**6.8**	**4.7**	**6.4**	**4.6**	2.2	**0.6**	**4.5**

Table 2. **Area under the curve of the absolute trajectory error.** COLMAP succeeds on textured objects like the first two but struggles to recover the complete trajectory for less textured objects.

Object	PSNR↑ / SSIM↑ / LPIPS↓	
	Ours	UNISURF [22]
3: cracker box	29.77 / 0.73 / 0.31	29.79 / 0.74 / 0.33
4: sugar box	30.77 / 0.82 / 0.31	30.73 / 0.76 / 0.33
6: mustard	30.73 / 0.74 / 0.39	30.72 / 0.74 / 0.37
10: potted meat	31.07 / 0.77 / 0.35	31.28 / 0.78 / 0.35
21: bleach	30.82 / 0.74 / 0.36	29.87 / 0.67 / 0.42
35: power drill	31.82 / 0.78 / 0.26	31.81 / 0.76 / 0.28
19: pitcher base	32.13 / 0.83 / 0.26	32.28 / 0.83 / 0.25
25: mug	31.18 / 0.74 / 0.39	31.69 / 0.76 / 0.37
Average	31.01 / 0.77 / 0.32	31.02 / 0.75 / 0.34

Table 3. **Evaluation of the estimated object texture.** The proposed method achieves comparable quality of the recovered object texture as UNISURF which uses ground-truth poses.

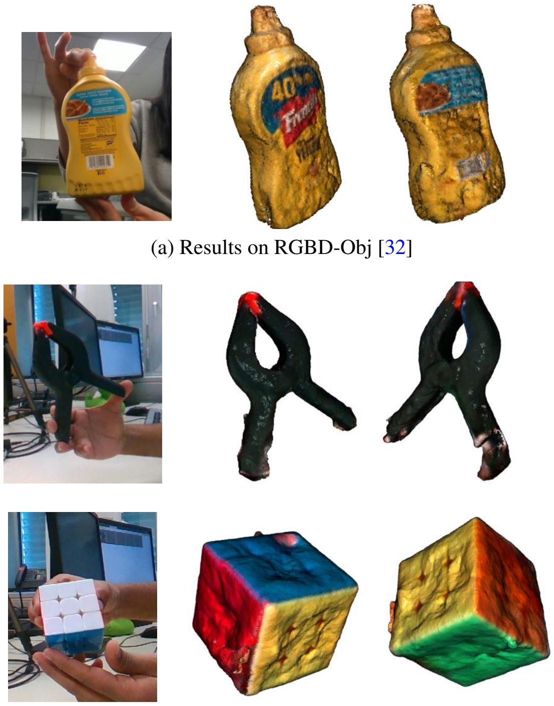
Seq. Name	w/o $\mathcal{L}_\text{d}$	w/o $\mathcal{L}_\text{f}$	w/o $\mathcal{L}_\text{r}$	All Terms
MDF14	4.9	4.8	1.1	**5.8**
SM2	1.0	2.9	0.5	**8.1**

Table 4. **Ablation study with AUC of the ATE metric.** All loss terms are required for obtaining accurate trajectories.

with the extra large clamp and Rubik's cube from YCB, which we captured for this project, are shown in Figure 9. Our method is able to produce 3D models also for the latter two objects, which are poorly textured and classical feature-based methods such as [29] fail to reconstruct them.

5.3. Ablation Study

The benefit of individual loss terms proposed in Sec. 3.4 is demonstrated qualitatively in Figures 3–5 and quantitatively in Table 4, where AUC of the ATE plot is shown for the largest segment for 2 sequences from the HO-3D dataset. We do not consider complete reconstruction in Table 4 as tracking fails completely for some segments without some of the loss terms. The optical flow loss enforces provides additional constraints on the predicted camera poses, the shape regularization loss stabilizes the optimization especially in its early stage, and the synthetic depth loss preserves previously reconstructed surface parts. Without the synthetic depth loss, the object can be significantly deformed especially when the camera performs larger motions in newly considered frames.



(a) Results on RGBD-Obj [32]

(b) Results on our new sequences

Figure 9. **Results on RGBD-Obj [32] and two new sequences.** The left column shows a sample image from the input sequence showing an unknown object manipulated by hands. The right column shows two views at the reconstructed color 3D model.

Figure 7 shows the importance of splitting the input sequence into segments. Figure 6 demonstrates the benefit of initializing the optimization of Eq. (5) with poses estimated from segments over initializing with random or zero poses.

6. Conclusion

We introduced a method that is able to reconstruct an unknown object manipulated by hands from color images. The main challenge resides in preventing drift during the simultaneous tracking and reconstruction. We believe our strategy of splitting the sequence based on the apparent area of the object and our regularization terms to be general and useful ideas, and hope they will inspire other researchers.

References

[1] Wout Boerdijk, Martin Sundermeyer, Maximilian Durner, and Rudolph Triebel. What's This?" - Learning to Segment Unknown Objects from Manipulation Sequences. In *International Conference on Robotics and Automation*, 2021. 3

[2] Mark Boss, Andreas Engelhardt, Abhishek Kar, Yuanzhen Li, Deqing Sun, Jonathan T. Barron, Hendrik P. A. Lensch, and Varun Jampani. SAMURAI: Shape And Material From Unconstrained Real-world Arbitrary Image Collections. In *Advances in Neural Information Processing Systems*, 2022. 3

[3] Javier Civera, Andrew J. Davison, and J. M. Martinez Montiel. Inverse Depth Parametrization for Monocular SLAM. In *International Conference on Computer Vision*, 2008. 6

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Conference on Computer Vision and Pattern Recognition*, 2009. 6

[5] Olivier D. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1993. 2

[6] Yasutaka Furukawa and Jean Ponce. Accurate, Dense, and Robust Multiview Stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8), 2009. 2

[7] Shreyas Hampali, Mahdi Rad, Markus Oberweger, and Vincent Lepetit. HOnnotate: A Method for 3D Annotation of Hand and Object Poses. In *Conference on Computer Vision and Pattern Recognition*, 2020. 2, 6

[8] Xian-Feng Han, Hamid Laga, and Mohammed Bennamoun. Image-Based 3D Object Reconstruction: State-Of-The-Art and Trends in the Deep Learning Era. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(5), 2019. 1

[9] Richard I. Hartley and Andrew Zisserman. Multiple Views Geometry in Computer Vision. *Cambridge University Press*, 2000. 1, 2

[10] Yana Hasson, Gül Varol, Dimitrios Tzionas, Igor Kalevatykh, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning Joint Reconstruction of Hands and Manipulated Objects. In *Conference on Computer Vision and Pattern Recognition*, 2019. 2

[11] Di Huang, Xiaopeng Ji, Xingyi He, Jiaming Sun, Tong He, Qing Shuai, Wanli Ouyang, and Xiaowei Zhou. Reconstructing hand-held objects from monocular video. In *SIGGRAPH Asia Conference Proceedings*, 2022. 2

[12] John F. Hughes, Andries Van Dam, Morgan Mcguire, David F. Sklar, James D. Foley, Steven Feiner, and Kurt Akeley. *Computer Graphics: Principles and Practice*. Addison-Wesley, 2013. 3

[13] Yoonwoo Jeong, Seokjun Ahn, Christopher Choy, Anima Anandkumar, Minsu Cho, and Jaesik Park. Self-Calibrating Neural Radiance Fields. In *International Conference on Computer Vision*, 2021. 2, 6

[14] Korrawe Karunratanakul, Jinlong Yang, Yan Zhang, Michael J. Black, Krikamol Muandet, and Siyu Tang. Grasping Field: Learning Implicit Representations for Human Grasps. In *International Conference on 3D Vision*, 2020. 2

[15] Zhengfei Kuang, Kyle Olszewski, Menglei Chai, Zeng Huang, Panos Achlioptas, and Sergey Tulyakov. NeROIC: Neural Rendering of Objects from Online Image Collections. *IEEE Transactions on Robotics and Automation*, 41(4), July 2022. 2

[16] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. BARF: Bundle-Adjusting Neural Radiance Fields. In *International Conference on Computer Vision*, 2021. 2, 4, 6

[17] William E. Lorensen and Harvey E. Cline. Marching Cubes: A High-Resolution 3D Surface Construction Algorithm. In *ACM SIGGRAPH*, 1987. 3

[18] Zhaoyang Lv, Edward Miller, Jeff Meissner, Luis Pesqueira, Chris Sweeney, Jing Dong, Lingni Ma, Pratik Patel, Pierre Moulon, Kiran Somasundaram, Omkar Parkhi, Yuyang Zou, Nikhil Raina, Steve Saarinen, Yusuf M Mansour, Po-Kang Huang, Zijian Wang, Anton Troynikov, Raul Mur Artal, Daniel DeTone, Daniel Barnes, Elizabeth Argall, Andrey Lobanovskiy, David Jaeyun Kim, Philippe Bouttefroy, Julian Straub, Jakob Julian Engel, Prince Gupta, Mingfei Yan, Renzo De Nardi, and Richard Newcombe. Aria pilot dataset. https://about.facebook.com/realitylabs/projectaria/datasets, 2022. 2

[19] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy Networks: Learning 3D Reconstruction in Function Space. In *Conference on Computer Vision and Pattern Recognition*, 2019. 1

[20] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *European Conference on Computer Vision*, 2020. 2, 3, 6

[21] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable Volumetric Rendering: Learning Implicit 3D Representations Without 3D Supervision. In *Conference on Computer Vision and Pattern Recognition*, 2020. 1, 2

[22] Michael Oechsle, Songyou Peng, and Andreas Geiger. UNISURF: Unifying Neural Implicit Surfaces and Radiance Fields for Multi-View Reconstruction. In *International Conference on Computer Vision*, 2021. 1, 2, 3, 7, 8

[23] Jeong Joon Park, Peter Florence, Julian Straub, Richard A. Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *Conference on Computer Vision and Pattern Recognition*, 2019. 1

[24] Timothy Patten, Kiru Park, Markus Leitner, Kevin Wolfram, and Markus Vincze. Object Learning for 6D Pose Estimation and Grasping from RGB-D Videos of In-Hand Manipulation. In *International Conference on Intelligent Robots and Systems*, 2021. 6, 7, 8

[25] David Prokhorov, Dmitry Zhukov, Olga Barinova, Konushin Anton, and Anna Vorontsova. Measuring Robustness of Visual SLAM. In *Machine Vision and Applications*, 2019. 6

[26] Martin Runz, Kejie Li, Meng Tang, Lingni Ma, Chen Kong, Tanner Schmidt, Ian Reid, Lourdes Agapito, Julian Straub,

Steven Lovegrove, and Richard Newcombe. Frodo: from Detections to 3D Objects. In *Conference on Computer Vision and Pattern Recognition*, 2020. 1

[27] Szymon Rusinkiewicz, Olaf Hall-Holt, and Marc Levoy. Real-Time 3D Model Acquisition. In *ACM SIGGRAPH*, 2002. 1, 2

[28] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition*, 2016. 7, 8

[29] Johannes L. Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise View Selection for Unstructured Multi-View Stereo. In *European Conference on Computer Vision*, 2016. 1, 2, 7, 8

[30] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo Tourism: Exploring Photo Collections in 3D. In *ACM SIGGRAPH*, 2006. 2

[31] Dimitrios Tzionas and Juergen Gall. 3D Object Reconstruction from Hand-Object Interactions. In *International Conference on Computer Vision*, 2015. 1, 2

[32] Fan Wang and Kris Hauser. In-Hand Object Scanning via RGB-D Video Segmentation. In *International Conference on Robotics and Automation*, 2019. 2, 6, 7, 8

[33] Pengyuan Wang, Fabian Manhardt, Luca Minciullo, Lorenzo Garattoni, Sven Meie, Nassir Navab, and Benjamin Busam. Demograsp: Few-shot learning for robotic grasping with human demonstration. *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5733–5740, 2021. 2

[34] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. NeRF--: Neural Radiance Fields Without Known Camera Parameters. In *arXiv Preprint*, 2021. 2

[35] Thibaut Weise, Bastian Leibe, and Luc Van Gool. Accurate and Robust Registration for In-Hand Modeling. In *Conference on Computer Vision and Pattern Recognition*, 2008. 1, 2

[36] Thibaut Weise, Thomas Wismer, Bastian Leibe, and Luc Van Gool. In-Hand Scanning with Online Loop Closure. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, 2009. 1, 2

[37] Thibaut Weise, Thomas Wismer, Bastian Leibe, and Luc Van Gool. Online Loop Closure for Real-Time Interactive 3D Scanning. *Computer Vision and Image Understanding*, 115(5), 2011. 2

[38] Junfeng Wu, Yi Jiang, Wenqing Zhang, Xiang Bai, and Song Bai. SeqFormer: A Frustratingly Simple Model for Video Instance Segmentation. In *arXiv Preprint*, 2021. 3

[39] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. *Science*, 2018. 6

[40] Gengshan Yang and Deva Ramanan. Volumetric correspondence networks for optical flow. In *NeurIPS*, 2019. 5

[41] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Huiwen Chang, Deva Ramanan, William T. Freeman, and Ce Liu. LASR: Learning Articulated Shape Reconstruction from a Monocular Video. In *Conference on Computer Vision and Pattern Recognition*, 2021. 6

[42] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume Rendering of Neural Implicit Surfaces. In *Advances in Neural Information Processing Systems*, 2021. 1, 2

[43] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview Neural Surface Reconstruction by Disentangling Geometry and Appearance. In *Advances in Neural Information Processing Systems*, 2020. 1, 2

[44] Yufei Ye, Abhinav Gupta, and Shubham Tulsiani. What's in Your Hands? 3D Reconstruction of Generic Objects in Hands. In *Conference on Computer Vision and Pattern Recognition*, 2022. 2, 7, 8

[45] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Phillip Krähenbühl, and Ishan Misra. Detecting Twenty-Thousand Classes Using Image-Level Supervision. In *European Conference on Computer Vision*, 2022. 3