Online 3D Scene Reconstruction Using Neural Object Priors

Thomas Chabal[†‡] Shizhe Chen[†‡] Jean Ponce[‡*] Cordelia Schmid[†‡]

Abstract

This paper addresses the problem of reconstructing a scene online at the level of objects given an RGB-D video sequence. While current object-aware neural implicit representations hold promise, they are limited in online reconstruction efficiency and shape completion. Our main contributions to alleviate the above limitations are twofold. First, we propose a feature grid interpolation mechanism to continuously update grid-based object-centric neural implicit representations as new object parts are revealed. Second, we construct an object library with previously mapped objects in advance and leverage the corresponding shape priors to initialize geometric object models in new videos, subsequently completing them with novel views as well as synthesized past views to avoid losing original object details. Extensive experiments on synthetic environments from the Replica dataset, real-world ScanNet sequences and videos captured in our laboratory demonstrate that our approach outperforms state-of-the-art neural implicit models for this task in terms of reconstruction accuracy and completeness.

1. Introduction

Reconstructing a scene from a moving camera is a fundamental task in computer vision, with a wide range of applications in augmented reality [51], robotics [1, 40], autonomous driving [27, 94], entertainment or cultural heritage preservation. Despite the availability of numerous approaches [58] to tackle the problem, it is still challenging to construct on the fly detailed scene representations made of separate object models with manageable storage and computation resources.

The prevailing methods in scene reconstruction typically treat the entire scene as a single entity [4, 10, 13, 30, 47, 50, 51, 63, 77, 80, 98], using for example voxel grids [10, 13, 50, 51], point clouds [16, 17], surfel maps [66, 87] or meshes [62] to represent the scene. Recently, neural implicit representations, *e.g.*, NeRFs [45], and Gaussian splatting [31] have given promising results in scene reconstruction [77, 98] thanks to their reduced memory footprint and enhanced accuracy at fine-grained resolutions. However, extracting individual objects from scene-level implicit representations proves to be time and computation intensive [3, 32], resulting in inflexibility to independently manipulate, modify or replace objects in the scene.

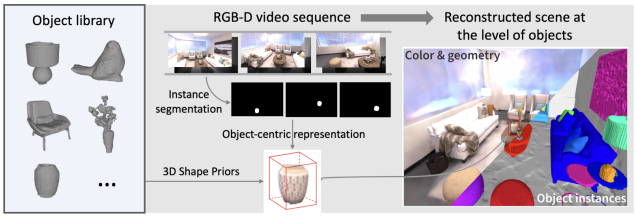

Figure 1. Our method reconstructs scenes at the level of objects from RGB-D videos on the fly. We leverage 3D shape priors from a pre-computed object library to enhance accuracy and completeness of geometry reconstruction for individual objects.

To address these limitations, there is a growing focus on object-centric neural implicit representations [22, 35] for scene reconstruction. vMAP [35] is a state-of-the-art approach in this direction that optimizes a separate multilayer perceptron (MLP) for each object. While vMAP is lightweight and operates in an online setting, its ability to reconstruct intricate object details is constrained by its simple MLP model. RO-MAP [22] enhances MLPs with multiresolution hash coding [47]. However, it requires to accumulate all recorded views of an object to estimate the object bounding box for optimization and, thus, sacrifices online efficiency for accuracy. Furthermore, the above approaches optimize models from scratch without leveraging 3D priors, thus focusing only on object parts visible in the current video and leading to slower and incomplete reconstructions. Yet, there is a clear benefit to having access to prior object shape knowledge, *i.e.*, we can better reconstruct an object if we have seen similar objects or the same object from different angles before. Most prior endeavours incorporating object priors either only estimate object poses and cannot reconstruct novel objects [37–39, 90] (see, however [64, 65]) or rely on a single latent vector in initialization [76, 99] which hardly captures fine-grained object shape and may lose details from past views.

In this work, we overcome the aforementioned limita-

[†] Inria. [‡] Department of Computer Science, École normale supérieure (ENS-PSL, CNRS, Inria). [*] Courant Institute of Mathematical Sciences and Center for Data Science, New York University. {firstname.lastname}@inria.fr

tions with object-centric neural implicit representations for online scene reconstruction. Given an RGB-D video, we obtain object masks and camera poses at each frame following prior work [35], then reconstruct all objects in the scene on the fly by optimizing the implicit object representation with differentiable volume rendering. Each object representation consists of a feature grid tightly encompassing the object of interest and a small MLP to predict occupancy and RGB colors. Geometry and appearance are disentangled to allow the optimization of one without deteriorating the other. To allow full online optimization as opposed to the partially online approach in [22], we continuously expand the feature grid and adapt our representation with feature interpolation as unseen parts of objects become visible. This enables us to start reconstructing objects as soon as they enter the camera's field of view. Moreover, we introduce a novel approach to leverage previously observed objects and enhance online optimization of object shapes. Specifically, we first construct an object library containing fitted object models from either full 3D meshes or previous mapping videos. If and when an object is identified in the video, we retrieve the most similar object from the library and align it to the current scene. If correctly matched, the shape representation of the retrieved object serves as initialization for the new one. To ensure keeping the geometric information in the prior object model, we propose to render images from the prior model as a surrogate for past observations and use them as additional signal when optimizing the current model. We conduct extensive experiments on synthetic environments from the Replica dataset, real-world ScanNet sequences and our own videos. Experimental results demonstrate the effectiveness of the proposed system in achieving more accurate and more complete object-aware scene reconstruction.

To summarize, our contribution is two-fold:

- We adapt an object-centric grid-based neural implicit representation that we equip with a feature grid interpolation scheme to incrementally reconstruct objects as soon as they become visible. We design it for flexible reuse of geometry across videos, high accuracy and efficiency of optimization.
- We leverage pre-constructed shape representations from completely or partially observed objects to accelerate shape optimization and enhance completeness. We address the problem of losing shape details from past views by rendering images from previous viewpoints as additional supervision.

The proposed system achieves state-of-the-art performance on both the synthetic Replica dataset, real environments in ScanNet and videos recorded in our laboratory. Our code and models are publicly released [1].

[1] https://github.com/thomaschabal/online-scene-reconstruction

2. Related Work

2.1. Online dense mapping

Reconstructing a 3D scene from a set of images is a long-standing problem in computer vision [10, 23]. A large branch of the community tackled it with approaches running in an offline fashion, with all the images initially given as input [2, 21, 23, 64, 72]. Working in an online setting, visual Simultaneous Localization And Mapping (vSLAM) estimates camera poses while mapping scenes from a stream of images, the two problems being tightly linked [79]. While the first real-time approaches considered sparse visual anchors [14, 33], later works improved them with better suited features [19, 49] or semi-dense point clouds [16, 17], improving performances while typically operating at frame-rate on a CPU. More recent works introduced learning-based approaches for monocular systems [11, 34, 78, 92, 93], but these require heavy computation and tend to not transfer well to scenes outside the training distribution. With the emergence of consumer-grade depth cameras, dense SLAM systems [13, 50, 51, 62, 73] have been developed, typically relying on truncated signed distance functions (TSDF) [10, 52], a voxel-based representation of the world that allows for fast incremental merge of depth views. While the memory usage of these grids grows cubically with the spatial resolution, some works [26] have explored the storage of latent codes in grids, decoded by a small neural network, to improve reconstruction with coarser grids. Other scene representations include point clouds [57] and surfels [66, 87] which are fast to compute but remain discrete, a limitation for a number of downstream applications. Few works tackle the problem of object-level SLAM [44, 91], possibly in dynamic scenes [66, 67], representing signed distance functions as either grids [44], octrees [91] or surfels [66, 67], but they do not leverage any prior on the objects they reconstruct.

2.2. NeRF-based SLAM

With the advent of differentiable volume rendering [45], localization and mapping approaches based on neural implicit models have been proposed [9, 22, 29, 30, 35, 63, 71, 77, 80, 98], benefiting from improvement in novel view synthesis [4, 20, 47]. Unlike traditional voxel grids, point clouds and meshes, implicit representations are light, continuous, differentiable and accurate at any resolution. These works have originally encoded the spatial position of a point with a small neural network [35, 77] before storing features in a point cloud [71] or on a sparse [29] or dense [30, 98] grid, possibly indexed by a hashing function [9, 22, 29, 63, 80]. A small MLP is then used as a decoder. If some systems follow the original NeRFs by modelling the world with density functions, this formulation does not explicitly define surfaces and has thus been replaced by signed distance functions [56, 83] or, rather equivalently, occupancy values [53] in later reconstruction approaches.

Prior NeRF-based works mainly consider the world as a single entity and process it with one single model [9, 29, 30, 63, 71, 77, 80, 98]. This makes it difficult to extract objects or reuse objects in different scenes, in particular for representations relying on hash grids or specific positional encodings. Furthermore, such models uselessly represent empty space in rooms. Some recent works further exploit instance segmentation masks to isolate and reconstruct objects in real-time [22, 35, 85]. BundleSDF [85] relies on an efficient hash grid representation [47, 52] but is restricted to reconstructing a single object. Conversely and closest to our work, vMAP [35] and RO-MAP [22] parallelize the optimization of multiple models, one per object. However, the former only relies on MLPs, leading to poor accuracy and slow computation time, while the latter exploits a combination of hash grids and MLPs but it requires objects to be fully in view to estimate their geometry. Other approaches extract accurate object reconstructions by modelling specifically the presence of objects when optimizing a single scene representation [88, 89], but they run offline. None of the methods presented here reuses any knowledge from outside the environment they are optimized on.

2.3. 3D reconstruction with object priors

Using object priors has been explored extensively in several fields of computer vision. As an object-level SLAM system, SLAM++ [70] exploits a database of few objects mapped offline to refine its localization, but does not additionally map novel objects in a scene. Another set of works reconstruct objects by learning latent codes for shapes or categories with large object databases [68, 74, 76, 82, 99] and optimizing one code per novel object given input views. However, they are unable to map objects of categories unseen at training and often fail to capture shape details visible in input views. Recent works also leverage diffusion models to generate realistic images for unseen viewpoints of an object [36, 60, 81], but these do not represent the actual unseen parts. In object pose estimation, numerous approaches have been proposed to estimate the 6D pose of an object in one or multiple observations of a scene given a 3D model of that object [24, 37–39, 55, 64, 65, 90]. Current approaches to that problem typically employ neural networks to estimate a coarse pose [24, 37, 55, 90] and then apply refinement by rendering the object [38, 39]. While these models fit in a reconstruction pipeline, they are usually restricted to estimating poses for the objects seen at training time [24, 37, 39, 90], require an accurate 3D model of each object to be available and cannot reconstruct novel objects.

3. Method

Given an RGB-D video V of a static scene captured by a moving camera C, our goal is to build object-centric representations for the scene that can accurately reconstruct the whole scene at the level of objects. We assume for that, as in [22, 35], that we have access at each frame to object masks tracked consistently in the whole video, one mask per object in the scene, and camera poses expressed in a fixed world frame w, both provided by external off-the-shelf systems, *e.g.*, [6, 7, 48, 49]. In the following, we first describe our object-centric model for online scene reconstruction and its optimization in Sec. 3.1. We then introduce in Section 3.2 the offline construction of an object library and how to leverage it to enhance the online shape optimization of the same objects observed in novel scenes with different viewpoints.

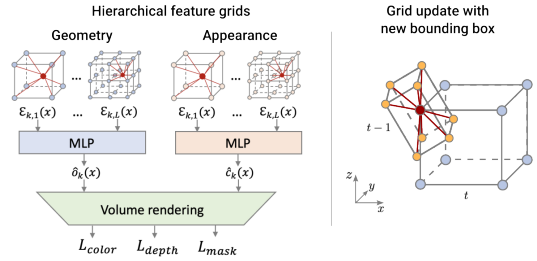

Figure 2. (Left) Our object-centric representation. Given a 3D point x inside the object bounding box, we predict its occupancy and color values via two small feature grids and MLPs. The object model is trained by volume rendering. (Right) Given the mapping between object bounding boxes at times $t-1$ and t, we retrieve features in the former feature grid to update the new one.

3.1. Online object-centric scene reconstruction

Object-centric representation. For any object O_k in the scene, we have its segmentation masks before the current time t. As the object has a compact surface, we can restrict its extent to a bounded volume. To this end, we store a low-resolution point cloud of O_k, denoted as $\mathcal{P}_{k,1:t}$, using object masks from the first time O_k appears in video V to time t. We then compute a bounding box encompassing $\mathcal{P}_{k,1:t}$, the center and orientation of which are used to define the frame attached to O_k. We denote the rigid transformation from the world frame w to the object frame at time t as $S_t^k = [s_t^k R_t^k, T_t^k]$ where s_t^k, R_t^k and T_t^k are respectively the box extent, orientation and position for object O_k at time t in the world frame. The point cloud $\mathcal{P}_{k,1:t}$, bounding box and matrix S_t^k are updated with time as more parts of O_k are seen in the video.

We represent O_k by a function mapping a 3D point in its bounding box to a color and an occupancy value, as illustrated on the left of Figure 2. Specifically, the function is composed of two separate grid encoders and MLP decoders, one predicting occupancy for the geometry and the other predicting colors for appearance. Each grid is a dense multi-resolution grid, similarly to [47] though not using a hash index, which impedes sequential optimization, and encoding only shape or color. It fully covers the bounding box of O_k, its boundaries being the ones of that bounding box. The grid stores features at L different levels. At level

$l = 1, ..., L$, it has $N_l = N_0 \gamma^{l-1}$ vertices per side, with N_0 the side size of the coarsest level and γ a scale factor, and each vertex stores a single feature scalar $E_{k,l}$ in a table. When querying the encoding of a 3D point x in the grid, we retrieve the stored features for the vertices of the cell x falls in at each level l, trilinearly interpolate them and concatenate the L resulting scalars. The output embedding $\mathcal{E}_k(x) \in \mathbb{R}^L$ is differentiable with respect to the stored features in $E_{k,l}$, which are optimized variables. We feed $\mathcal{E}_k(x)$ to an MLP which predicts either an occupancy value $\hat{o}_k(x)$ as in [53] or a color $\hat{c}_k(x)$. Surfaces are defined as the $\frac{1}{2}$-level set of the occupancy function. Note that, following prior work [30, 35, 71, 77, 98], we do not model the color dependency on the viewing direction.

Updating feature grids with new views. As the feature grid for object O_k is closely aligned with the bounding box of the object, it must be adapted to cover the extended bounding box with new views. Specifically, given object frames at previous time S^k_{t-1} and the current time S^k_t, we first compute the mapping between the previous and current bounding box $\Delta S_{k,t-1,t} = (S^k_{t-1})^{-1} S^k_t$. We then map each vertex v of the new grid to the former bounding box using $\Delta S_{k,t-1,t}$ and retrieve the corresponding interpolated feature in the former grid, which we store as the feature for v in the new grid. The right part of Figure 2 illustrates the process. We assign random values to vertices that fall out of the previous bounding box. In this way, the interpolated feature grid replaces the previously stored one to allow incremental updating. The optimization continues with new feature grid parameters.

Training objectives with volume rendering. We employ differentiable volume rendering for training. First, during the mapping sequence, we store a buffer of 20 keyframes as well as the two running frames for each object O_k as in vMAP [35]. At each time t, we randomly select a set of frames $\mathcal{F}_{k,t}$ in this buffer, and sample pixels that are part of the masks of O_k. Then, given a camera pose T and a pixel u, we sample N increasing depth values on the ray $T \cdot u$, among which N_{surf} values follow a normal distribution centered at the measured depth value $D(u)$ with a manually set variance σ^2 and the other $N - N_{surf}$ values are uniformly distributed between the bounding box's closest border to the camera and $D(u) - 3\sigma$. Having a bounding box around each object leads to a more efficient sampling by not querying many points in empty space, unlike vMAP [35]. These points are then expressed in the object frame and fed to the density function to get occupancy and color values $\hat{o}_{k,i}$ and $\hat{c}_{k,i}$. We compute the ray termination weight at the i-th point on a ray as $w_{k,i} = \hat{o}_{k,i} \prod_{j<i}(1 - \hat{o}_{k,j})$ and obtain the pixel color $\hat{C}_k$, depth $\hat{D}_k$, mask $\hat{M}_k$ and depth variance $\hat{V}_k$ of object O_k with classical volume rendering (see formulas in the supplementary material).

We optimize the model with cost functions penalizing the difference between the inputs and the renderings for color $\mathcal{L}_{col}$, depth $\mathcal{L}_{depth}$ and mask $\mathcal{L}_{mask}$ for each pixel u (detailed in the supplementary material):

$$\mathcal{L} = \frac{1}{Z} \sum_{k=1}^{K} \sum_{u \in \mathcal{F}_{k,t}} \mathcal{L}_{depth} + \lambda_{col} \mathcal{L}_{col} + \lambda_{mask} \mathcal{L}_{mask}. \quad (1)$$

where K is the number of objects and $Z = K|\mathcal{F}_{k,t}|$. Note here that each object is independent from the others in the final loss, allowing their optimizations to be parallelized.

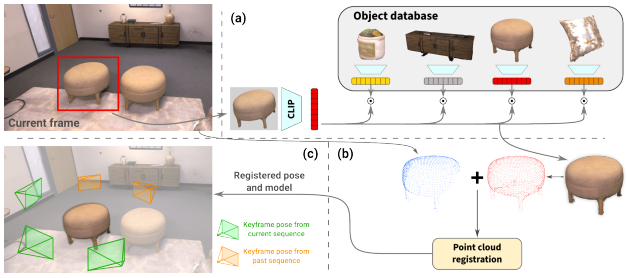

Figure 3. Overview of the procedure to integrate prior object models. *(a)* Retrieval: given a newly segmented object, we retrieve the most similar object in the object library via CLIP embedding. *(b)* Registration: we get an aligned pose of the retrieved object via point cloud registration. *(c)* Shape refinement: we refine the initial shape model with novel views while additionally synthesizing keyframes from retrieved object models to not lose shape details.

3.2. Integrating prior object-centric neural shape models

In order to enhance accuracy and shape completion and benefit from knowledge accumulated during previous mapping sessions, we propose to leverage 3D shape priors for online object reconstruction. We construct an object library containing a set of N objects $\{M_1, ..., M_N\}$ that have already been mapped, be it during previous mapping episodes or with images coming from 3D meshes. For each object, we store its neural implicit representation (shape feature grid and MLP), a coarse point cloud, and features for matching. During new mapping sessions, we first search for objects in the library that may appear in each frame and compute initial poses. If matches are found, we then texture and complete the initial shapes if necessary, taking a special care in not losing details seen only during previous episodes.

Building an object library. We consider two types of resources to obtain objects for our object library. The first type is from an existing database of 3D meshes, which is for instance relevant in industrial environments where some objects have already been precisely scanned (*e.g.*, [25]) and their geometry can be reused with no further updating. While this database can be of any size, our initialization procedure, which we present next, focuses on reusing the exact same object as the one in the current view, bringing little interest in using large-scale databases with most objects never observed, unlike other works [68, 74, 76, 82, 99]. We generate synthetic photo-realistic RGB-D images and segmentation masks for each object, taken at multiple random

poses, using the BlenderProc renderer [15]. These images almost fully cover the complete object. The second type is from previous mapping sessions, *e.g.*, another video stream capturing the same scene, or one with objects shared with the current scene, from different angles. This scenario is practical in real world applications where a robot enters a room several times to reconstruct it, not necessarily with a 360° scan, stores object models at the end of each video and then relocates and completes them with novel viewpoints, a setting similar to [84]. Compared to objects in the first type, the objects here are likely to only be partially observed. We use the same method described in Sec. 3.1 to reconstruct objects from images. Therefore, each object M_j in the library contains its appearance and shape feature grids and MLPs, as well as its bounding box extent, a low-resolution point cloud which we obtain by rendering and backprojecting depth maps from the object model, and the poses in the object frame of keyframes stored during the model acquisition in previous sequences. Note that storing and optimizing geometry separately from appearance is possible thanks to their disentanglement into two feature grids and MLPs, as explained above. We further extract global and local features for each object which are later used for retrieval and registration. For the retrieval part, we render RGB images with the model from some stored keyframe poses and compute CLIP embeddings [61] for each view. We store the averaged embedding of all the views as the single retrieval vector. For registration, we compute normals in the stored point cloud and extract and store FPFH features [69].

Retrieval and registration. During a new sequence, when a novel object O_k is seen in a frame, we identify the most similar object in our object database and align it with the current view. First, we crop the current RGB frame around O_k, replace the background by some fixed color and compute a CLIP embedding for this image. We retrieve the $m = 3$ objects in our database that have the same semantic category and highest cosine similarity to O_k and filter these matches based on a threshold on their matching scores. We then register the retrieved point clouds with the backprojected current partial view of the object thanks to Ransac [18] followed by a point-to-plane ICP [5] relying on normals and likewise filter fitness scores larger than another threshold. We additionally reproject the registered point clouds in the camera view and ensure that it coincides well with the object mask and that all the points are further away from the camera than the input depth map. If these conditions are filled, we consider these retrieval and registration to be successful, M_j and O_k being then assumed to be the same object. We finally initialize O_k as M_j, using the registered pose as the object pose in the world frame, and fix the MLPs, and optionally the feature grids, for the rest of the sequence. If the cosine similarity or fitness thresholds are not reached, we start reconstructing O_k from scratch and retry this initialization when more of the object is visible, *i.e.*, when the bounding box is grown one more time.

Synthesizing keyframes on the fly. Initializing an object from a previous model and updating it with solely novel views as explained previously gradually loses the shape information from the original model. Yet, storing and reusing all past keyframes is infeasible for long sequences with numerous objects, which is why we only maintain a buffer of keyframes per object. To alleviate this issue, we instead synthesize on the fly views from the retrieved model and add them to the keyframe buffer. Inspired by previous work [59], at each optimization step and for each initialized object, we sample $\frac{|\mathcal{F}_{k,t}|}{2}$ camera poses among those stored in the database and render color, depth and mask from these poses using a fixed copy of the reused model. These synthesized views are particularly important for object parts not seen in the current video. Our experiments show that this allows to complete the object with novel views while preserving the original shape details.

4. Experiments

4.1. Experimental setting

Datasets. We evaluate the proposed approach on two datasets of indoor scenes, Replica [75, 77] and ScanNet [12], and on our own sequences. The original Replica dataset consists of synthetic, noiseless 2000-frame RGB-D videos, one for each of 8 synthetic environments, along with the corresponding camera poses and segmentation masks [75]. Ground-truth meshes are also available for quantitative evaluation. To get more meaningful evaluations, we manually remove from this dataset ground-truth meshes with fewer than 50 vertices, *e.g.*, product tags, and clean a few noisy meshes. The ScanNet dataset consists of real RGB-D videos of various durations recorded with a moving tablet. It also comes with camera poses from BundleFusion [13] as well as manually annotated object masks which however are still noisy. Our own sequences are recorded in our lab with a Realsense D435 RGB-D camera filming static scenes of few objects. We compute camera poses with ORB-SLAM2 [48] and extract and track consistent masks with Tracking-Anything [7, 41].

Implementation details. For our object models, the feature grid contains $L = 3$ levels with $N_0 = 16$ and $\gamma = 1.5$, and the MLPs consist respectively of 1 and 2 64-neuron hidden layers for geometry and color. As this work focuses on object representations, we use the background model from vMAP [35], which is a single MLP. For our object library, we compare two resources to get object models: one using 3D meshes, which are the ground-truth Replica meshes extracted by vMAP [35]; the other using another video sequence acquired with different viewpoints for each environment, which was also generated by vMAP [35]. We

	Object prior	Seen parts Acc.↓	Comp.↓	Whole objects Acc.↓	Comp.↓	CR 1cm↑	CR 5mm↑
TSDF* [10]	—	**0.61**	0.38	**0.59**	3.07	73.7	69.2
vMAP* [35]	—	1.32	0.79	2.31	2.10	73.0	52.7
Ours	—	0.82	0.34	2.31	2.43	81.3	75.9
	3D meshes	0.78	**0.26**	1.25	**0.29**	**98.7**	**93.9**
	Prior video	0.81	0.33	2.28	1.36	85.2	80.1

Table 1. Object-level reconstruction performance averaged over 8 Replica scenes evaluated with seen parts and whole objects as ground-truth meshes. Our results with object priors rely on ground truth meshes and shape from previously viewed videos. CR stands for Completion Ratio. Methods with * are reproduced results from the official code bases.

use the ground-truth retrieval and registration in the experiments if not otherwise mentioned. In optimization, we set $\lambda_{mask} = 10$ and take $\lambda_{color} = 5$ on Replica and 2 on ScanNet. We provide additional details about our implementation in the supplementary material.

Metrics. We measure the quality of reconstructions on meshes extracted from the Replica sequences using *accuracy* and *completion* in *cm* as well as *completion ratio* with various thresholds, as in prior work [35]. Denoting the reconstructed and ground-truth meshes as R and G, accuracy is defined as the average distance between points in R and their nearest neighbour in G and completion is the average distance between points in G and their nearest neighbour in R. We consider two types of ground-truth meshes in our evaluation. The first is so-called seen parts where we cull the original mesh by removing vertices that never appear in the input video. We focus on the accuracy metric for the seen parts. The second is the whole mesh including both seen and unseen parts from the video. It is more meaningful to measure the completion metrics with the whole mesh when using object priors that inform about parts unseen in the current video. In addition, unlike prior works [30, 35, 71, 77, 98] that subsample vertices on the ground-truth meshes for evaluation, we consider all the vertices to remove any impact from this sampling. Computation time is also measured and discussed in the supplementary material. For the ScanNet dataset, since no ground truth mesh exist and object masks are very noisy, our evaluations are thus only qualitative.

Baselines. We compare our method with prior object-level reconstruction methods. We run all the compared methods with their released codes and the provided ground-truth camera poses for fair comparison. vMAP [35] and RO-MAP [22] are the only object-level neural implicit RGB-D methods we are aware of. However, since RO-MAP did not release codes on the evaluated datasets and details lack in the paper to reproduce faithfully their results, we mainly compare with vMAP in the main paper and put the discussion of RO-MAP in the supplementary material. We also evaluate an object-level TSDF [10] integration.

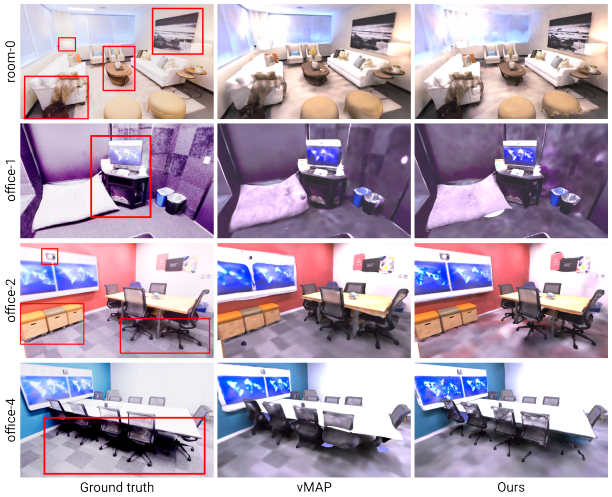

Figure 4. Examples of reconstructions with our method on different Replica scenes, compared to vMAP [35]. Our method recovers object geometry that is more faithful to the actual shapes and with better texture.

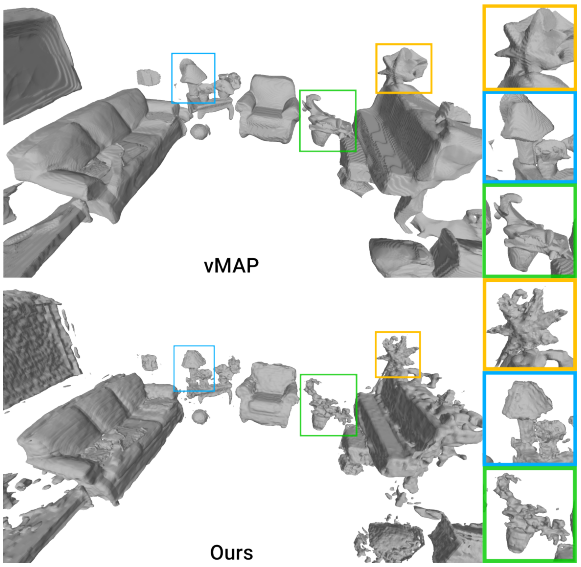

Figure 5. Reconstruction of a ScanNet sequence with vMAP and our method, with close-up views on some parts. Our method recovers more accurate geometries than vMAP, which over-smoothes surfaces - see in particular the piano and plant on the right or the sofa on the left - though it is a bit more sensitive to ScanNet's noisy inputs.

Most neural implicit RGB-D methods operate at the level of scenes, *e.g.* [30, 71, 77]: we also compare to them in the supplementary material. We extract object meshes at a resolution of 5mm for all methods using marching cubes [42], and disable any mesh post-processing. Each method is run with 5 different initializations on each environment, and we present results averaged over all runs.

4.2. Comparison with the state of the art

Replica. Table 1 presents the object-level reconstruction performance of vMAP [35], TSDF [10] and variants of

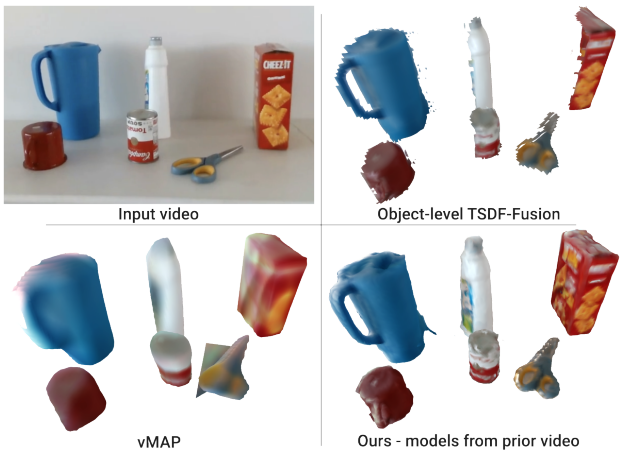

Figure 6. Reconstruction of a self-captured sequence with object-level TSDF, vMAP and our method using a prior video.

our method. When no object shape priors are considered in both approaches, our method outperforms vMAP across all metrics for the seen parts, with a relative improvement of around 40% in accuracy and 60% in completion. This highlights the superior capability of our object-centric neural implicit model in faithfully reconstructing objects from video streams. For metrics on the whole objects, we observe a decrease in completion for our method. This discrepancy arises since vMAP uses pure MLPs to predict smooth occupancy values in space, then extracting regular surfaces beyond the seen parts. In contrast, while the feature grid we employ is more powerful, it is also more local and only predicts surfaces for the visible parts of objects. However, completion ratios are much higher with our model than vMAP on whole meshes, with an increase of 23.2% at 5mm, showcasing superior capability to accurately fit seen ground-truth mesh parts. Compared to TSDF, our approach is less accurate on seen parts and whole meshes, but is more complete and has higher completion ratios. As our method relies on MLPs and multi-scale continuous feature grids, it may slightly benefit from some surface regularity beyond seen parts, unlike TSDF, explaining this gap. Adding object priors mitigates this issue and leads to a substantial improvement in completion metrics for whole object evaluation. When compared to using full 3D meshes as shape priors, using object models fitted on prior video sequences results in a decrease in performance. This can be attributed to the fact that the two video sequences for each scene cover very similar viewpoints of objects, resulting in limited potential improvement on these metrics. We show a few examples of our reconstructions on scenes of the Replica dataset in Figure 4. Our object representations recover geometry that is more accurate and faithful to the original scene.

ScanNet. For real-world scenes, we show a reconstructed scene from the ScanNet dataset in Figure 5. Again, our reconstructions are closer to the actual geometry of the scene than vMAP. However, our approach is more sensitive to

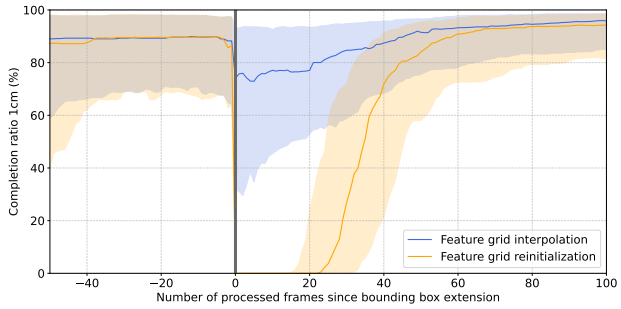

Figure 7. Evolution of the median completion ratio at 1cm before and after object bounding box extensions, either interpolating grid features or reinitializing the grid from scratch. Colored areas represent the 20-th and 80-th percentiles. Interpolating features leads to much smaller decays and more precise meshes right after box extension (vertical line).

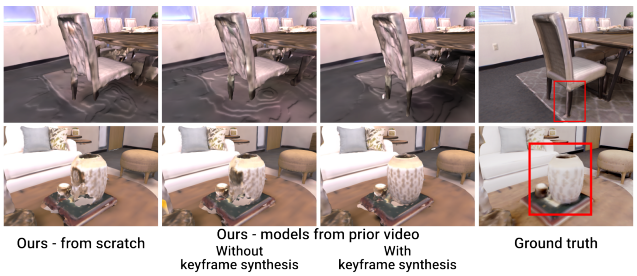

Figure 8. Visualization of completed meshes with our method on Replica scenes. Interesting parts are highlighted. Note that the back of the chair in the first row is never seen in any video, which explains its poor reconstruction.

noise in the input segmentation masks and depth images (see supplementary material for noisy input data). Since our bounding boxes are computed from these inputs and grown as soon as a single segmented point for the object falls outside it, an ill-segmented image may have the boxes grow too much and lead to artifacts in areas that are not or little mapped, for which stored features are random. Our object grid representations also optimize local features unlike vMAP's MLPs whose weights are shared for a whole object and thus more global. This implies that our models are optimized locally to fit all points including outliers, whereas vMAP oversmoothes geometries, trading a loss of important object details for a higher robustness to outliers.

Our own video. We further evaluate the reconstruction of a real-world sequence acquired in our lab, where we compare variants of our model to TSDF [10] and vMAP [35]. The results are shown in Figure 6. Again, our base model recovers much finer details than vMAP and has fewer floating artifacts and more regular geometries on parts seen in few frames than TSDF. When we additionally leverage a library made from another sequence of the same objects captured with different viewpoints, we obtain higher completion for the objects while preserving details for parts that were only seen in the past video.

Object library	Keyframe synthesis	Seen parts Acc.↓	Seen parts Comp.↓	Whole mesh Acc.↓	Whole mesh Comp.↓	Whole mesh CR 1cm↑	Whole mesh CR 5mm↑
3D meshes	Fixed models	0.77	**0.23**	**1.21**	**0.26**	**99.3**	**97.1**
	✗	**0.76**	0.34	1.96	1.61	82.8	76.4
	✓	0.78	0.26	1.25	0.29	98.7	93.9
Prior video	✗	0.80	0.34	2.16	1.82	82.5	77.0
	✓	0.81	0.33	2.28	1.36	85.2	80.1

Table 2. Ablations of the keyframe synthesis part of our method, using different object libraries. Metrics are computed at the level of objects and retrieval and registration are ground truth. In the first row, the whole object models are frozen.

4.3. Ablations

Feature grid interpolation. We compare two strategies for the grid update at each bounding box extension: reinitializing the grid features [22] or interpolating them, as proposed above. For that, we compute completion ratios at 1cm on seen parts of objects after each frame of each Replica sequence. We then select the metrics in a window of few frames before and after each box extension and average them over all the objects and their box update steps. The result is shown on Figure 7. We observe that reinitializing features requires 20 more frames after the update before extracting a surface from the representation. Conversely, interpolating features results in a small decay of the metric immediately after the interpolation step, a value that is otherwise reached after fitting for 40 more frames when reinitializing the features. This shows the importance of our interpolation strategy to get an accurate model at all time.

Fitting with synthesized keyframes. We evaluate in Table 2 the effectiveness of the proposed strategy to avoid losing original shape details when utilizing the shape prior. In the upper block of the table, as the reused object models are obtained from full 3D meshes, simply freezing them achieves high performance in both accuracy and completion. However, continuing to update them with no further change, as indicated in the second row, exhibits a significant drop in completion especially on the whole mesh. This underlines the severity of the loss of detail problem. The proposed strategy plays a crucial role in addressing it for unseen parts, maintaining comparable performance to the unoptimized models, see the third row. When employing object models fitted on prior video sequences in the bottom block, updating the models, imperfect due to partial observations, becomes necessary. In this case, our proposed strategy preserves similar performance on seen parts without compromising completion metric on the whole mesh. Our strategy thus allows to complete partially seen objects while not deteriorating fully mapped ones. As determining whether an object has already been fully seen can be difficult, we demonstrate here that all objects can be fitted within a single framework. Figure 8 shows examples of using object priors from video sequences with and without keyframe synthesis.

Object library	GT retr. & reg.	Seen parts Acc.↓	Seen parts Comp.↓	Whole mesh Acc.↓	Whole mesh Comp.↓	Whole mesh CR 1cm↑	Whole mesh CR 5mm↑
—	✗	0.82	0.34	2.31	2.43	81.3	75.9
3D meshes	✓	**0.78**	**0.26**	**1.25**	**0.29**	**98.7**	**93.9**
	✗	0.84	0.45	1.97	1.94	86.4	79.9
Prior video	✓	0.81	0.33	2.28	1.36	85.2	80.1
	✗	0.85	0.88	2.34	2.77	81.3	75.3

Table 3. Ablations of the retrieval and registration parts of our method, using different object libraries. Metrics are computed at the level of objects. The first row is the reconstruction from scratch. CR stands for Completion Ratio.

Object retrieval and registration. Table 3 assesses the influence of object retrieval and registration. Though utilizing ground-truth retrieval and registration enhances performance compared to the method without any object priors, the automatic retrieval and registration method notably degrades overall performance, though leading to more accurate and complete whole meshes when using 3D meshes priors. In detail, objects are correctly retrieved in 57% and 78% for the full 3D mesh and previous video setups respectively, and 22% and 51% of objects are both accurately retrieved and registered. When retrieval or registration fails, our models are then fitted from scratch, similarly to the first row. A few large objects are initialized with a wrong pose with our retrieval and registration, explaining the decrease in completion from the last row. Future work improving automatic retrieval and registration will benefit our method.

5. Conclusion

We present an online method to reconstruct scenes at the level of objects from RGB-D video sequences. Leveraging object masks and camera poses obtained from the video, we adapt neural implicit object models consisting of grid-based occupancy and color fields, which can be incrementally updated with new views of the object. To further improve reconstruction accuracy and completeness, we build object libraries from prior sequences or available meshes and propose a way to utilize them as shape prior and update the pre-trained object models without forgetting. Experiments on Replica and ScanNet datasets as well as real-world sequences demonstrate the effectiveness of our approach. Limitations of our work and future directions are discussed in the supplementary material.

Acknowledgment

This work was performed using HPC resources from GENCI-IDRIS (Grants 2021-AD011012725R1/R2). It was funded in part by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute). JP was supported in part by the Louis Vuitton/ENS chair in artificial intelligence and a Global Distinguished Professorship at the Courant Institute of Mathematical Sciences and the Center for Data Science at New York University.

References

[1] Michal Adamkiewicz, Timothy Chen, Adam Caccavale, Rachel Gardner, Preston Culbertson, Jeannette Bohg, and Mac Schwager. Vision-only robot navigation in a neural radiance world. *IEEE Robotics Autom. Lett.*, 7(2):4606–4613, 2022. 1

[2] Dejan Azinovic, Ricardo Martin-Brualla, Dan B. Goldman, Matthias Nießner, and Justus Thies. Neural RGB-D surface reconstruction. In *CVPR*, pages 6280–6291. IEEE, 2022. 2

[3] Kenneth Blomqvist, Francesco Milano, Jen Jen Chung, Lionel Ott, and Roland Siegwart. Neural implicit vision-language feature fields. *CoRR*, abs/2303.10962, 2023. 1

[4] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *ECCV (32)*, pages 333–350. Springer, 2022. 1, 2

[5] Yang Chen and Gérard G. Medioni. Object modelling by registration of multiple range images. *Image Vis. Comput.*, 10(3):145–155, 1992. 5, 4

[6] Ho Kei Cheng and Alexander G. Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *ECCV (28)*, pages 640–658. Springer, 2022. 3

[7] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Alexander G. Schwing, and Joon-Young Lee. Tracking anything with decoupled video segmentation. In *ICCV*, pages 1316–1326. IEEE, 2023. 3, 5

[8] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. *arXiv preprint arXiv:2212.07143*, 2022. 4

[9] Chi-Ming Chung, Yang-Che Tseng, Ya-Ching Hsu, Xiang Qian Shi, Yun-Hung Hua, Jia-Fong Yeh, Wen-Chin Chen, Yi-Ting Chen, and Winston H. Hsu. Orbeez-slam: A real-time monocular visual SLAM with ORB features and nerf-realized mapping. In *ICRA*, pages 9400–9406. IEEE, 2023. 2, 3

[10] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, pages 303–312. ACM, 1996. 1, 2, 6, 7, 3, 5, 9, 11

[11] J Czarnowski, T Laidlow, R Clark, and AJ Davison. Deepfactors: Real-time probabilistic dense monocular slam. *IEEE Robotics and Automation Letters*, 5:721–728, 2020. 2

[12] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, pages 2432–2443. IEEE Computer Society, 2017. 5, 11

[13] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Trans. Graph.*, 36(3):24:1–24:18, 2017. 1, 2, 5

[14] Andrew J. Davison, Ian D. Reid, Nicholas Molton, and Olivier Stasse. Monoslam: Real-time single camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6):1052–1067, 2007. 2

[15] Maximilian Denninger, Dominik Winkelbauer, Martin Sundermeyer, Wout Boerdijk, Markus Knauer, Klaus H. Strobl, Matthias Humt, and Rudolph Triebel. Blenderproc2: A procedural pipeline for photorealistic rendering. *Journal of Open Source Software*, 8(82):4901, 2023. 5, 4, 13

[16] Jakob Engel, Thomas Schöps, and Daniel Cremers. LSD-SLAM: large-scale direct monocular SLAM. In *ECCV (2)*, pages 834–849. Springer, 2014. 1, 2

[17] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *CoRR*, abs/1607.02565, 2016. 1, 2

[18] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. 5, 4

[19] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza. SVO: semidirect visual odometry for monocular and multicamera systems. *IEEE Trans. Robotics*, 33(2):249–265, 2017. 2

[20] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, pages 5491–5500. IEEE, 2022. 2

[21] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multi-view stereopsis. In *CVPR*. IEEE Computer Society, 2007. 2

[22] Xiao Han, Houxuan Liu, Yunchao Ding, and Lu Yang. Ro-map: Real-time multi-object mapping with neural radiance fields. *IEEE Robotics and Automation Letters*, 8(9): 5950–5957, 2023. 1, 2, 3, 6, 8, 7

[23] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004. 2

[24] Rasmus Laurvig Haugaard and Anders Glent Buch. Surfemb: Dense and continuous correspondence distributions for object pose estimation with learnt surface embeddings. In *CVPR*, pages 6739–6748. IEEE, 2022. 3

[25] Tomas Hodan, Frank Michel, Eric Brachmann, Wadim Kehl, Anders Glent Buch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, Caner Sahin, Fabian Manhardt, Federico Tombari, Tae-Kyun Kim, Jiri Matas, and Carsten Rother. BOP: benchmark for 6d object pose estimation. In *ECCV (10)*, pages 19–35. Springer, 2018. 4

[26] Jiahui Huang, Shi-Sheng Huang, Haoxuan Song, and Shi-Min Hu. Di-fusion: Online implicit 3d reconstruction with deep priors. In *CVPR*, pages 8932–8941. Computer Vision Foundation / IEEE, 2021. 2

[27] Yuanhui Huang, Wenzhao Zheng, Yunpeng Zhang, Jie Zhou, and Jiwen Lu. Tri-perspective view for vision-based 3d semantic occupancy prediction. In *CVPR*, pages 9223–9232. IEEE, 2023. 1

[28] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021. If you use this software, please cite it as below. 4

[29] Chenxing Jiang, Hanwen Zhang, Peize Liu, Zehuan Yu, Hui Cheng, Boyu Zhou, and Shaojie Shen. H2-mapping: Real-time dense mapping using hierarchical hybrid representation. *CoRR*, abs/2306.03207, 2023. 2, 3

[30] Mohammad Mahdi Johari, Camilla Carta, and François Fleuret. ESLAM: efficient dense SLAM system based on hybrid representation of signed distance fields. In *CVPR*, pages 17408–17419. IEEE, 2023. 1, 2, 3, 4, 6, 5, 9

[31] Bernhard Kerbl, Georgios Kopanas, and Thomas Leimkühlerand George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (SIGGRAPH)*, 42, 2023. 1

[32] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. LERF: language embedded radiance fields. *CoRR*, abs/2303.09553, 2023. 1

[33] Georg Klein and David William Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR*, pages 225–234. IEEE Computer Society, 2007. 2

[34] Lukas Koestler, Nan Yang, Niclas Zeller, and Daniel Cremers. TANDEM: tracking and dense mapping in real-time using deep multi-view stereo. In *CoRL*, pages 34–45. PMLR, 2021. 2

[35] Xin Kong, Shikun Liu, Marwan Taher, and Andrew J. Davison. vmap: Vectorised object mapping for neural field SLAM. *CVPR*, 2023. 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

[36] Xin Kong, Shikun Liu, Xiaoyang Lyu, Marwan Taher, Xiaojuan Qi, and Andrew J. Davison. Eschernet: A generative model for scalable view synthesis. *CoRR*, abs/2402.03908, 2024. 3

[37] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *ECCV (17)*, pages 574–591. Springer, 2020. 1, 3

[38] Yann Labbé, Lucas Manuelli, Arsalan Mousavian, Stephen Tyree, Stan Birchfield, Jonathan Tremblay, Justin Carpentier, Mathieu Aubry, Dieter Fox, and Josef Sivic. Megapose: 6d pose estimation of novel objects via render & compare. In *CoRL*, pages 715–725. PMLR, 2022. 3

[39] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *ECCV (6)*, pages 695–711. Springer, 2018. 1, 3

[40] Peiqi Liu, Yaswanth Orru, Chris Paxton, Nur Muhammad (Mahi) Shafiullah, and Lerrel Pinto. Ok-robot: What really matters in integrating open-knowledge models for robotics. *CoRR*, abs/2401.12202, 2024. 1

[41] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: marrying DINO with grounded pre-training for open-set object detection. *CoRR*, abs/2303.05499, 2023. 5

[42] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *SIGGRAPH*, pages 163–169. ACM, 1987. 6, 1, 2, 3, 7

[43] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017. 4

[44] John McCormac, Ronald Clark, Michael Bloesch, Andrew J. Davison, and Stefan Leutenegger. Fusion++: Volumetric object-level SLAM. In *3DV*, pages 32–41. IEEE Computer Society, 2018. 2

[45] 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 *ECCV (1)*, pages 405–421. Springer, 2020. 1, 2

[46] Thomas Müller. tiny-cuda-nn, 2021. 4

[47] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. 1, 2, 3

[48] Raul Mur-Artal and Juan D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Trans. Robotics*, 33(5):1255–1262, 2017. 3, 5

[49] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. Robotics*, 31(5):1147–1163, 2015. 2, 3

[50] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew W. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *ISMAR*, pages 127–136. IEEE Computer Society, 2011. 1, 2

[51] Richard A. Newcombe, Steven Lovegrove, and Andrew J. Davison. DTAM: dense tracking and mapping in real-time. In *ICCV*, pages 2320–2327. IEEE Computer Society, 2011. 1, 2

[52] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Trans. Graph.*, 32(6):169:1–169:11, 2013. 2, 3

[53] Michael Oechsle, Songyou Peng, and Andreas Geiger. UNISURF: unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *ICCV*, pages 5569–5579. IEEE, 2021. 2, 4

[54] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision. *Trans. Mach. Learn. Res.*, 2024, 2024. 1

[55] Evin Pinar Örnek, Yann Labbé, Bugra Tekin, Lingni Ma, Cem Keskin, Christian Forster, and Tomas Hodan. Foundpose: Unseen object pose estimation with foundation features. *CoRR*, abs/2311.18809, 2023. 3, 1

[56] Joseph Ortiz, Alexander Clegg, Jing Dong, Edgar Sucar, David Novotný, Michael Zollhöfer, and Mustafa Mukadam. isdf: Real-time neural signed distance fields for robot perception. *CoRR*, abs/2204.02296, 2022. 2

[57] Matia Pizzoli, Christian Forster, and Davide Scaramuzza. REMODE: probabilistic, monocular dense reconstruction in real time. In *ICRA*, pages 2609–2616. IEEE, 2014. 2

[58] Julio A Placed, Jared Strader, Henry Carrillo, Nikolay Atanasov, Vadim Indelman, Luca Carlone, and José A

Castellanos. A survey on active simultaneous localization and mapping: State of the art and new frontiers. *IEEE Transactions on Robotics*, 2023. 1

[59] Ryan Po, Zhengyang Dong, Alexander W. Bergman, and Gordon Wetzstein. Instant continual learning of neural radiance fields. In *ICCV (Workshops)*, pages 3326–3336. IEEE, 2023. 5

[60] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*. OpenReview.net, 2023. 3

[61] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PMLR, 2021. 5, 4

[62] Antoni Rosinol, Andrew Violette, Marcus Abate, Nathan Hughes, Yun Chang, Jingnan Shi, Arjun Gupta, and Luca Carlone. Kimera: From SLAM to spatial perception with 3d dynamic scene graphs. *Int. J. Robotics Res.*, 40(12-14): 1510–1546, 2021. 1, 2

[63] Antoni Rosinol, John J. Leonard, and Luca Carlone. Nerf-slam: Real-time dense monocular SLAM with neural radiance fields. *CoRR*, abs/2210.13641, 2022. 1, 2, 3

[64] Fred Rothganger, Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *International Journal ofComputer Vision*, 66: 231–259, 2006. 1, 2, 3

[65] Fred Rothganger, Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Segmenting, modeling, and matching video clips containing multiple moving objects. *IEEE transactions on pattern analysis and machine intelligence*, 29(3): 477–491, 2007. 1, 3

[66] Martin Rünz and Lourdes Agapito. Co-fusion: Real-time segmentation, tracking and fusion of multiple objects. In *ICRA*, pages 4471–4478. IEEE, 2017. 1, 2

[67] Martin Rünz, Maud Buffier, and Lourdes Agapito. Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. In *ISMAR*, pages 10–20. IEEE, 2018. 2

[68] Martin Rünz, Kejie Li, Meng Tang, Lingni Ma, Chen Kong, Tanner Schmidt, Ian D. Reid, Lourdes Agapito, Julian Straub, Steven Lovegrove, and Richard A. Newcombe. Frodo: From detections to 3d objects. In *CVPR*, pages 14708–14717. Computer Vision Foundation / IEEE, 2020. 3, 4

[69] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (FPFH) for 3d registration. In *ICRA*, pages 3212–3217. IEEE, 2009. 5, 4

[70] Renato F. Salas-Moreno, Richard A. Newcombe, Hauke Strasdat, Paul H. J. Kelly, and Andrew J. Davison. SLAM++: simultaneous localisation and mapping at the level of objects. In *CVPR*, pages 1352–1359. IEEE Computer Society, 2013. 3

[71] Erik Sandström, Yue Li, Luc Van Gool, and Martin R. Oswald. Point-slam: Dense neural point cloud-based SLAM. *CoRR*, abs/2304.04278, 2023. 2, 3, 4, 6, 9

[72] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[73] Thomas Schöps, Torsten Sattler, Christian Häne, and Marc Pollefeys. 3d modeling on the go: Interactive 3d reconstruction of large-scale scenes on mobile devices. In *3DV*, pages 291–299. IEEE Computer Society, 2015. 2

[74] Mo Shan, Qiaojun Feng, You-Yi Jau, and Nikolay Atanasov. ELLIPSDF: joint object pose and shape optimization with a bi-level ellipsoid and signed distance function description. In *ICCV*, pages 5926–5935. IEEE, 2021. 3, 4

[75] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Yuheng Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard A. Newcombe. The replica dataset: A digital replica of indoor spaces. *CoRR*, abs/1906.05797, 2019. 5, 2, 6, 8, 9, 10

[76] Edgar Sucar, Kentaro Wada, and Andrew Davison. Nodeslam: Neural object descriptors for multi-view shape reconstruction. In *2020 International Conference on 3D Vision (3DV)*, pages 949–958. IEEE, 2020. 1, 3, 4

[77] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J. Davison. imap: Implicit mapping and positioning in real-time. In *ICCV*, pages 6209–6218. IEEE, 2021. 1, 2, 3, 4, 5, 6, 9

[78] Zachary Teed and Jia Deng. DROID-SLAM: deep visual SLAM for monocular, stereo, and RGB-D cameras. In *NeurIPS*, pages 16558–16569, 2021. 2

[79] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT Press, 2005. 2

[80] Hengyi Wang, Jingwen Wang, and Lourdes Agapito. Co-slam: Joint coordinate and sparse parametric encodings for neural real-time SLAM. In *CVPR*, pages 13293–13302. IEEE, 2023. 1, 2, 3

[81] Hengyi Wang, Jingwen Wang, and Lourdes Agapito. Morpheus: Neural dynamic 360deg surface reconstruction from monocular rgb-d video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20965–20976, 2024. 3

[82] Jingwen Wang, Martin Rünz, and Lourdes Agapito. DSP-SLAM: object oriented SLAM with deep shape priors. In *3DV*, pages 1362–1371. IEEE, 2021. 3, 4

[83] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *NeurIPS*, pages 27171–27183, 2021. 2

[84] David Watkins-Valls, Peter K. Allen, Henrique Maia, Madhavan Seshadri, Jonathan Sanabria, Nicholas R. Waytowich, and Jacob Varley. Watkins-valls, mobile manipulation leveraging multiple views, iros 2022. In *IROS*, pages 4585–4592. IEEE, 2022. 5

[85] Bowen Wen, Jonathan Tremblay, Valts Blukis, Stephen Tyree, Thomas Müller, Alex Evans, Dieter Fox, Jan Kautz, and Stan Birchfield. Bundlesdf: Neural 6-dof tracking and 3d reconstruction of unknown objects. *CVPR*, 2023. 3

[86] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *CVPR*, pages 17868–17879. IEEE, 2024. 1

[87] Thomas Whelan, Stefan Leutenegger, Renato F. Salas-Moreno, Ben Glocker, and Andrew J. Davison. Elasticfusion: Dense SLAM without A pose graph. In *Robotics: Science and Systems*, 2015. 1, 2

[88] Qianyi Wu, Xian Liu, Yuedong Chen, Kejie Li, Chuanxia Zheng, Jianfei Cai, and Jianmin Zheng. Object-compositional neural implicit surfaces. In *ECCV (27)*, pages 197–213. Springer, 2022. 3

[89] Qianyi Wu, Kaisiyuan Wang, Kejie Li, Jianmin Zheng, and Jianfei Cai. Objectsdf++: Improved object-compositional neural implicit surfaces. In *ICCV*, pages 21707–21717. IEEE, 2023. 3

[90] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In *Robotics: Science and Systems*, 2018. 1, 3

[91] Binbin Xu, Wenbin Li, Dimos Tzoumanikas, Michael Bloesch, Andrew J. Davison, and Stefan Leutenegger. Mid-fusion: Octree-based object-level multi-instance dynamic SLAM. In *ICRA*, pages 5231–5237. IEEE, 2019. 2

[92] Nan Yang, Rui Wang, Jörg Stückler, and Daniel Cremers. Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. In *ECCV (8)*, pages 835–852. Springer, 2018. 2

[93] Nan Yang, Lukas von Stumberg, Rui Wang, and Daniel Cremers. D3VO: deep depth, deep pose and deep uncertainty for monocular visual odometry. In *CVPR*, pages 1278–1289. Computer Vision Foundation / IEEE, 2020. 2

[94] Ze Yang, Yun Chen, Jingkang Wang, Sivabalan Manivasagam, Wei-Chiu Ma, Anqi Joyce Yang, and Raquel Urtasun. Unisim: A neural closed-loop sensor simulator. In *CVPR*, pages 1389–1399. IEEE, 2023. 1

[95] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas A. Funkhouser. 3dmatch: Learning local geometric descriptors from RGB-D reconstructions. In *CVPR*, pages 199–208. IEEE Computer Society, 2017. 2, 3, 5, 11

[96] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, pages 11941–11952. IEEE, 2023. 1

[97] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing. *CoRR*, abs/1801.09847, 2018. 4

[98] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R. Oswald, and Marc Pollefeys. NICE-SLAM: neural implicit scalable encoding for SLAM. In *CVPR*, pages 12776–12786. IEEE, 2022. 1, 2, 3, 4, 6

[99] Zi-Xin Zou, Shi-Sheng Huang, Tai-Jiang Mu, and Yu-Ping Wang. Objectfusion: Accurate object-level slam with neural object priors. *Graphical Models*, 123:101165, 2022. 1, 3, 4

Online 3D Scene Reconstruction Using Neural Object Priors

Supplementary Material

The supplementary material is organized as follows: we discuss limitations of our method and possible directions to explore in Sec. 6, provide additional comparisons with state-of-the-art methods both quantitatively and qualitatively in Sec. 7 and introduce more implementation details in Sec. 8.

6. Limitations

We have identified three main limitations of our method. First, our models are highly sensitive to the quality of input masks, which define object bounding boxes extension and may severely worsen reconstructions. Correcting masks online from multiple views and the fitted models should robustify the whole model. Second, our retrieval and registration often struggle when seeing objects from too different viewpoints from those used for the database models. Further research on this part, *e.g.* using stronger image feature models for efficient retrieval [54, 96] and 6D object pose estimation for registration [55, 86], should allow for more systematic object reuse and reduce computations. Third, our method focuses on mapping objects given camera poses provided by an external SLAM system without correcting them. Extending it to a complete object-level SLAM is an interesting future direction.

7. Additional comparisons with the state of the art

Comparison to RO-MAP. We evaluate here our method in the setting of RO-MAP [22]. While the RO-MAP [22] paper does not provide all the details about their evaluations and no code has been released to reproduce results on the Replica dataset, we reproduce the closest setting to their work for fair comparison, following discussion with RO-MAP authors. In this way, we extract meshes for each object using marching cubes [42] on a grid of size 64^3 as detailed in their paper. We evaluate two scenes with the same objects, *i.e.*, 23 objects for the scene *room-0* and 14 for *office-1* as shared by RO-MAP authors. We present our quantitative results in Table 4 and qualitative examples of a scene and close-ups on few objects in Fig. 9 and Fig. 10. The meshes presented here for RO-MAP were shared by the authors. Our evaluations show that our models computed without any object prior are more accurate and have a better completion ratio at 1cm than RO-MAP, though the metric of completion distance is not so good. However, as shown in Fig. 9, RO-MAP objects possess numerous artefacts due to their uniform sampling of points along rays and different losses used, which explains their high accuracy error, very low completion distance and high completion ratio at 5cm, though large parts of objects are unseen in the input videos. In contrast, our object models are sampled only close to the surface during fitting, leading to much fewer outliers. Our method also extracts more faithful geometry on contours of objects as shown on Fig. 10. Leveraging object priors significantly improves the completion of objects in our method. While RO-MAP accumulates numerous views before computing a bounding box and restarts optimization from scratch each time these boxes change, our interpolation strategy allows us to reconstruct objects as soon as we observe them, even very partially, and then update their models at each frame without any reinitialization.

	Object prior	Scene	Whole objects Acc. ↓	Comp. ↓	CR 1cm ↑	CR 5cm ↑
RO-MAP† [22]	—	room-0	3.65	0.93	69.3	98.5
		office-1	3.74	1.15	67.9	97.7
Ours	—	room-0	2.04	2.02	73.8	90.0
		office-1	2.34	1.32	74.6	95.3
	3D meshes	room-0	1.31	0.58	86.2	99.9
		office-1	1.27	0.57	86.0	100.0
	Prior video	room-0	2.09	1.75	74.9	91.2
		office-1	2.43	1.11	76.4	96.3

Table 4. Object-level reconstruction performance compared to RO-MAP [22], using the scenes and settings of RO-MAP's evaluations. Our results with object priors rely on full 3D meshes and shapes from previously viewed videos. Retrieval and registration in these cases are ground truth. CR stands for Completion Ratio. Results for RO-MAP are taken from the original paper.

	Object prior	Whole objects Acc. ↓	Comp. ↓	CR 1cm ↑	CR 5cm ↑
vMAP† [35]	—	2.23	1.44	69.2	94.6
vMAP* [35]	—	1.84	2.32	63.6	91.5
Ours	—	1.52	2.58	73.9	91.0
	3D meshes	1.00	0.61	86.4	99.7
	Prior video	1.54	1.62	77.1	93.3

Table 5. Object-level reconstruction performance compared to vMAP [35], reproducing as closely as possible the evaluation setting of vMAP. Our results with object priors rely on full 3D meshes and shapes from previously viewed videos. Retrieval and registration in these cases are ground truth. CR stands for Completion Ratio. Results with † are taken from the original paper and with * are reproduced. The validity of the difference between the published and reproduced results for vMAP was confirmed by the authors of vMAP.

Comparison to vMAP. We further compare our method with vMAP on the same objects, metrics and evaluation pro-

	Objects	Object prior	Seen parts Acc. ↓	Seen parts Comp. ↓	Whole mesh Acc. ↓	Whole mesh Comp. ↓	Whole mesh CR 5cm ↑	Whole mesh CR 1cm ↑
TSDF* [10]	✗	—	0.55	0.41	2.66	4.31	87.0	81.8
iMAP* [77]	✗	—	0.92	0.91	1.89	2.42	90.1	74.7
ESLAM* [30]	✗	—	0.69	0.59	0.71	4.33	86.1	76.3
Point-SLAM* [71]	✗	—	0.67	0.59	0.67	4.76	85.3	79.5
vMAP* [35]	✓	—	1.03	0.91	2.85	2.49	91.2	75.0
Ours	✓	—	0.81	0.72	2.96	2.39	91.5	78.3
	✓	3D meshes	0.82	0.70	2.08	1.71	95.3	86.3
	✓	Prior video	0.80	0.70	2.69	2.25	92.3	79.3

Table 6. Averaged scene-level reconstruction metrics on the Replica dataset [75], focusing on the ground truth mesh parts observable in the input sequence (left) and whole scene (right). Our results with object priors rely on full 3D meshes and shapes from previously viewed videos. CR stands for Completion Ratio. Methods with * are reproduced results from the official code bases.

cedure as used by vMAP [35], to provide fair comparison to their published results and confirm that our updated setup proposed in the main paper does not artificially improve our performances over other baselines. Differing from the evaluations of the main paper, we consider here all objects in each scene, regardless of their size or presence of noise. We still extract object meshes at a 5mm resolution but set a maximum of 256 points per size and subsample 10k points in the GT and reconstructed meshes to compute metrics instead of considering full meshes. Results are presented in Tab. 5. For vMAP, we provide results taken from the original paper as well as reproduced results from the released code. Though these reproduced results differ from the published ones, we confirmed the validity of this difference with the main authors of vMAP. Reconstruction results with our models also confirm the trend observed in the main paper with our other evaluation setup: our models optimized from scratch are again more accurate than vMAP and have a higher completion ratio at 1cm. Their performances are similarly further increased by leveraging object shape priors. We finally present in Fig. 11 few visualizations of object meshes extracted after only 50 frames processed in the input sequence. In that setting, vMAP [35] tends to produce oversmoothed meshes which lack geometric and appearance details, *e.g.*, the texture of wood on the first row or the cushion's colored leaves on the last row. Conversely, our model from scratch recovers more faithful geometry and early texture for these objects. Adding object shape and texture prior further boosts the representations, leading to more complete and better textured models, in particular for our models obtained from a previous video.

Scene-level comparisons. For scene-level comparisons, our baselines are TSDF [10] (or rather its reimplementation [95]) with a grid resolution of 1cm, vMAP [35] and its reimplementation of iMAP [77], ESLAM [30] and Point-SLAM [71]. Again, we run all the compared methods with their released codes and the provided ground-truth camera poses for fair comparison. We extract scene meshes at a resolution of 1cm for all methods using marching cubes [42], and disable any mesh post-processing. Each method is run with 5 different initializations on each environment, and we present results averaged over all runs.

Results on the Replica dataset are presented in Table 6. As we reuse vMAP's background model, our method outperforms vMAP on the scene-level as well, aligning with the observed trend in object-level evaluation. Despite the improvement we achieve in object-level methods, our best object-level method with separate models per object still lags behind the best scene-level methods that consider scenes as a single entity when considering metrics on the seen parts or accuracy on whole meshes. Since the scene metrics depend heavily on the background quality, our background model which oversmoothes surfaces does not capture well details and explains this gap. Using a more accurate background model would benefit our method. However, our models present better completion distance and ratios on whole meshes than scene-level baselines, showing the interest of decomposing scenes in objects and reusing priors. In addition, it is worth noting that all the neural implicit model-based approaches perform worse on seen parts than the traditional TSDF [10], although they showcase advantages on the whole mesh.

We show meshes reconstructed by each of these methods on a Replica scene in Figure 12. The ground truth mesh is displayed here only for the parts that are seen in the input sequence. Methods relying on TSDF representations, *i.e.*, TSDF [10, 95] and Point-SLAM for its mesh extraction [71], are highly accurate for both objects and the background, though they do not reuse any prior information about the scene and are therefore unable to fill unseen parts. iMAP [77] and vMAP [35] extract oversmooth surfaces with some plausible completion for all objects and for the background, but they miss details of all reconstructed objects. ESLAM, which relies on a single tri-plane representation for the whole scene, proposes some completion for unseen parts of objects and for the background, but it misses important details about thin structures like pouf feet or the basket on the ground (left of the image). While our background model has limited ability to capture scene details, we obtain the highest level of details for all objects in the scene while being able to leverage prior knowledge to complete some parts, see for instance the back of poufs in the last image. We believe that further improvements in the background representation should make our method a strong competitor for online scene-level reconstructions.

Additional visualizations on Replica. As textures may prevent the reader from observing the geometry details of a mesh, we provide a textureless version of Figure 4 from the main paper in Fig. 13. These textureless images emphasize on the higher level of details recovered by our method com-

pared to vMAP [35]. In particular, thin structures, *e.g.*, table and chair feet or handles, are more challenging for vMAP but are correcly recovered by our approach.

Object meshes on ScanNet. We show some reconstructed objects on the ScanNet dataset in Fig. 14, using our method as well as vMAP [35] and a TSDF [10, 95] implemented with object grids of 5mm resolution. All meshes for these visualizations are extracted at a 1cm resolution using marching cubes [42]. Unlike other methods, we provide TSDF with knowledge of each object extent before starting the reconstruction since it is a static representation. Our method, run with no object prior, is able to reconstruct object geometries that are more accurate than vMAP [35] on these real world sequences. However, these sequences have very noisy depth and segmentation masks, resulting in artefacts in TSDF's reconstructions for ScanNet and slightly noisier ones for our method. The update time for TSDF representations also grows significantly with the resolution of the grid and number of objects, making the access to finer reconstructions much more costly than coarse resolutions, unlike our representations which keep a constant computation speed for any object resolution.

Additional results on sequences captured in our laboratory. We show in Figure 15 additional reconstructions on a scene with 3 objects, with views from the front and the back of the objects. As in the main paper, TSDF reconstructs objects with some floating artefacts around their borders. This objects are incomplete for parts unseen in the current video. vMAP outputs more complete but inaccurate object geometry with oversmoothed texture. Conversely, our approach produces more faithful shapes and textures for both seen and unseen parts of objects, in particular thanks to object initializations from a prior video.

Computation time. On Replica's room-0, our average times per frame are 740ms for objects reconstructed from scratch and 1.4s for models reused from the library. On the same hardware, vMAP and iMAP take 420ms per frame, ESLAM takes 445ms, Point-SLAM needs 32s and the scene-level TSDF runs at 18ms per frame. Our implementation is however not yet parallelized, unlike vMAP, which would yield important time savings. Important time savings can be obtained in several ways. First, improving the implementation to parallelize the fitting of all object models instead of optimizing them sequentially should lead to significant speed gains. Second, our retrieval and registration (R&R) currently operate in the same thread as model optimization, which stops at each R&R attempt, and require around 200ms per retrieval and registration attempt. Performing this stage in a separate thread should allow model optimization to run faster. Third, as object models are fitted separately, their optimization can be paused after converging on the currently stored keyframes. This is particularly useful for objects leaving the field of view for a long time, with no new stored keyframe. In this case, fewer objects are optimized, which benefits both computation time and GPU memory. Pausing the optimization is not feasible when considering the scene as a single entity. Note that these times do not include segmentation and mask tracking times, which are assumed to be run separately.

8. Implementation details

8.1. Object-centric model and optimization

Sampling points. At each frame and for each object, we perform 3 successive optimization steps, sampling 9600 rays among 6 keyframes for each step and 14 points per ray, 13 close to the surface and 1 closer to the camera. As explained in Section 3.1 of the main paper, the surface sampling consists in drawing points close to the surface on rays u following a normal distribution centered at the depth measurement $\mathcal{N}(D(u), \sigma)$. We take $3\sigma = 5cm$ for Replica scenes and a larger $\sigma = 10cm$ on ScanNet. The latter was observed to give better reconstructions due to ScanNet's noisy depth measurements and outliers that grow excessively object bounding boxes, representing large empty spaces.

Bounding boxes. We compute our bounding boxes using axis-aligned boxes in the world frame defined for each scene. Such bounding boxes are more efficient to compute than randomly aligned ones, though they may encompass larger empty space. When updating a bounding box, we add a 10% margin to the box extent in order to avoid doing this update too often as parts of objects are discovered at each frame. For each object, we only consider frames for which the object mask contains at least 100 pixels to avoid updating models based on too few observations.

Keyframe criterion. We reuse the same keyframe criterion as vMAP [35], which consists in considering every 25-th frame as a keyframe for objects and every 50-th for the background. We store keyframes in a buffer of up to 20 keyframes for both objects and background. Storing keyframes represents the largest memory usage of our approach, our object feature grids consisting in only around 65k parameters for shape and appearance respectively.

Volume rendering details We provide here more details about the rendering formulas and losses used for the online reconstruction. For each ray u, we compute ray termination

weights $w_{k,i}$ at each point as:

$$w_{k,i} = \hat{o}_{k,i} \prod_{j=1}^{i-1} (1 - \hat{o}_{k,j}), \tag{2}$$

We then render the pixel color $\hat{C}_k$, depth $\hat{D}_k$, mask $\hat{M}_k$ and depth variance $\hat{V}_k$ of object O_k as:

$$\hat{C}_k(u) = \sum_{i=1}^{N} w_{k,i} \hat{c}_{k,i}, \quad \hat{D}_k(u) = \sum_{i=1}^{N} w_{k,i} d_i, \tag{3}$$

$$\hat{M}_k(u) = \sum_{i=1}^{N} w_{k,i}, \quad \hat{V}_k(u) = \sum_{i=1}^{N} w_{k,i} (d_i - \hat{D}_k(u))^2. \tag{4}$$

For each object O_k, the losses used during fitting penalize the difference between the inputs and the renderings:

$$\mathcal{L}_{col}(k,u) = M_k(u) \| C(u) - \hat{C}_k(u) \|_1, \tag{5}$$

$$\mathcal{L}_{depth}(k,u) = M_k(u) \frac{\| D(u) - \hat{D}_k(u) \|_1}{\sqrt{\hat{V}_k(u)}}, \tag{6}$$

$$\mathcal{L}_{mask}(k,u) = \| M_k(u) - \hat{M}_k(u) \|_1, \tag{7}$$

where M_k is the binary mask of O_k.

Optimization details. We implement our feature grids and MLP using the tiny-cuda-nn library [46]. Our object models are optimized with AdamW [43] with learning rates 5×10^{-3} and 3.5×10^{-4} respectively for the feature grids and MLPs, and weight decay 0.1 for both. For the background model, we reuse the same parameters as the vMAP paper [35].

8.2. Integrating object shape priors

Constructing the object library. As explained in Section 3.2 of the main paper, we build object models offline from either full 3D meshes or video sequences. We detail here the first case. For each object 3D mesh, we render 40 images from random viewpoints around the object using the BlenderProc [15] renderer, each image being of size 1024×1024. We show examples of these renders in Fig. 16. 3D meshes for the Replica dataset are ground-truth object meshes and have been obtained by extracting a closed surface from a single volumetric representation. The latter extraction of object meshes performed by vMAP [35] consists in splitting this closed surface in objects according to vertex instance Ids, resulting in all objects being open surfaces. Thus, if camera poses are randomly sampled all around an object for our renders, the same part of a surface may be observed from two opposite viewpoints, which in turn causes problems during reconstruction. To avoid that issue, we use normal information to only retain one side of the surface. From these rendered images, we fit an object model with the method explained in Section 3.1 of the main paper, with the only difference that all inputs are known at the beginning of the fitting. Hence, we do not need to perform the online optimization but instead sample all frames at each optimization step. We perform 500 optimization steps per object and store the model at the last step for our database.

Retrieval and registration. For retrieval, we use the CLIP [61] version *ViT-bigG-14* from OpenClip [8, 28] and filter out retrieved objects for which the cosine similarity score is larger than 0.7. For the registration part, the FPFH [69] features, Ransac [18] and point-to-plane ICP [5] algorithms are reused from Open3D's implementation [97]. We filter the fitness with a threshold of 0.8 and keep registered poses for which at least 90% of reprojected points in the camera frame belong to the input object mask and have a depth larger than the depth measurement, with a tolerance of $2cm$.

Synthesizing keyframes on the fly. Once an object model has been initialized from the object library, we use the retrieved model to render additional views to fit the current object model. In this way, we sample half of the camera poses at each optimization step among the poses stored in the object library. We then render color, depth and mask using 24 points per ray, which we sample uniformly in the whole bounding box. This sampling contains more points than the one from current views (*i.e.*, 14 points) to cope with the absence of depth information that would otherwise guide the sampling around the actual object surface.

8.3. Evaluation datasets

Replica. Objects considered for the evaluations of the main paper slightly differ from those used in vMAP [35]. We clean few noisy meshes to remove vertices that are outliers and discard a few other tiny objects, *i.e.*, with fewer than 50 vertices. Examples of such objects are shown on Figure 17. Following this cleaning, Replica scenes contain on average 50 objects each.

ScanNet. For the ScanNet dataset, we additionally preprocess each depth image to remove outliers. More specifically, at each frame and for each object O_k, we compute the mean m_k and standard deviation s_k of depth values falling in the object mask and discard points whose depth is outside the range $[m_k - \alpha s_k, m_k + \alpha s_k]$, where we choose $\alpha = 1.5$, making this interval close to the 90% confidence interval of normal distributions. We also compute a histogram of depth points belonging to mask k with 15 values in the camera depth range ($[0m, 6m]$) and keep only bins that contain at

least 5% of points. This removes a large number of outliers, though some remain that may have a strong impact on our object bounding boxes. Further joint pre-processing of depth maps and segmentation masks would benefit our reconstructions on real world images. For fair comparison in the real world reconstructions, we also apply this pre-processing to TSDF [10, 95] and vMAP [35].

Real-world sequences. For our videos, we apply the same depth image processing as for ScanNet sequences and additionally erode object masks by few pixels to remove outliers and obtain better geometry.

8.4. Evaluation metrics

For evaluation on seen parts of meshes, we first cull vertices that are not seen in any input frame. We reuse ESLAM's culling script [30] and remove points at a depth $D_{rec} > D_{input} + \tau$, where D_{rec} is the depth of reprojected mesh points in camera frames, D_{input} is the measured depth for that frame and τ is a tolerance. We choose $\tau = 3cm$ for the scene meshes and $2cm$ for object meshes which we found to be a good trade-off between keeping all seen reconstructed vertices that may be inaccurately positioned and discarding all unseen ones. When evaluating reconstructions on the whole objects, including parts that are not seen in the input video sequence, we consider the full ground truth meshes, without applying this culling operation.

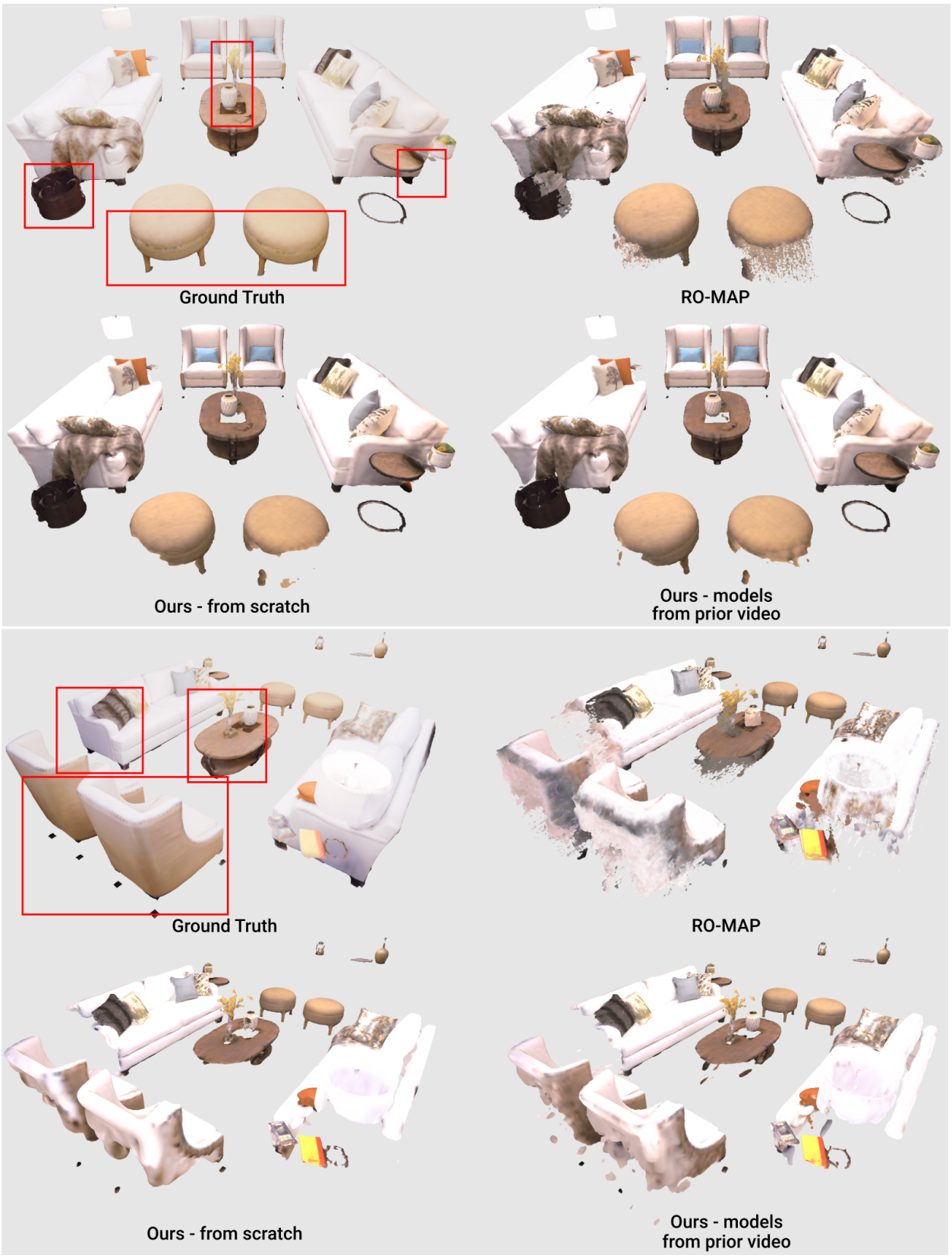

Figure 9. Comparison of object meshes from Replica [75] obtained by RO-MAP [22] and our method, using two different viewpoints. Meshes for RO-MAP were provided by the authors. Our meshes are extracted using marching cubes [42] on a grid of size 64^3 following RO-MAP's methodology and are restricted to objects reconstructed by RO-MAP. Regions of interest are highlighted in red.

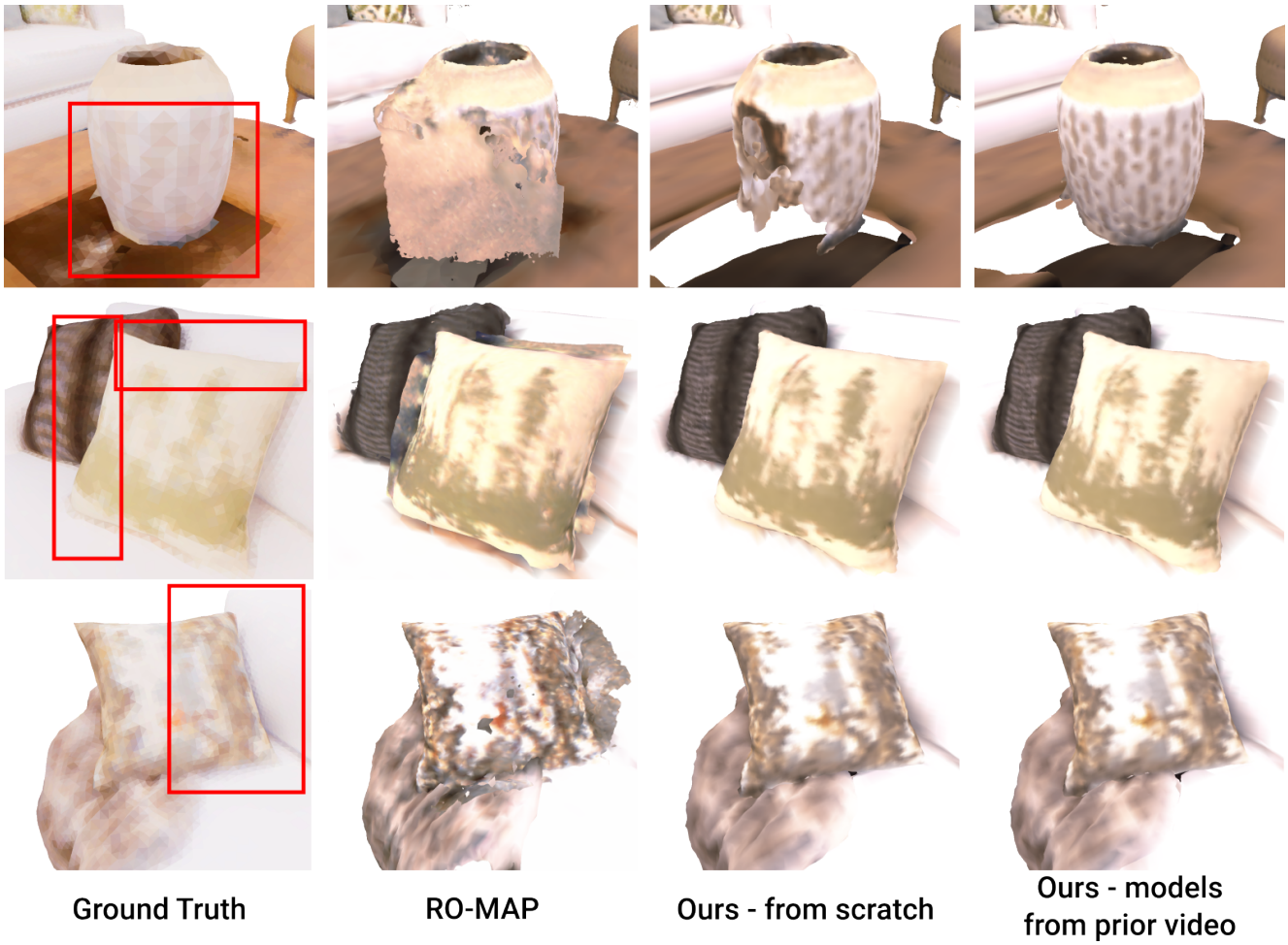

Figure 10. Comparison of object meshes obtained by RO-MAP [22] and our method. Meshes for RO-MAP were provided by the authors. Our meshes are extracted using marching cubes [42] on a grid of size 64^3 following RO-MAP's methodology. Regions of interest are highlighted in red. Note that the back of the vase on the first row is never seen in the first input sequence and only mapped in the second video which we use for our model shown in the last column.

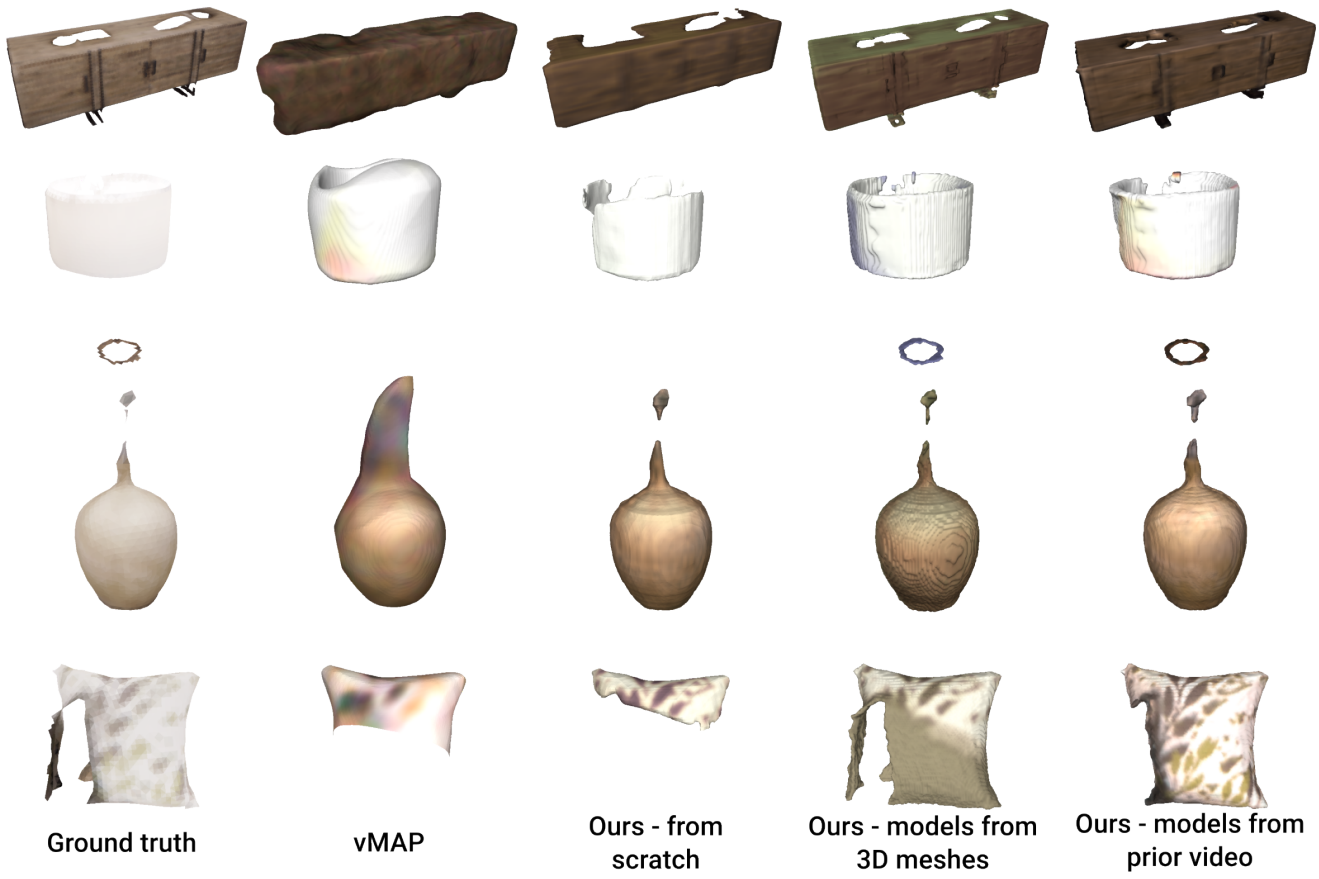

Figure 11. Comparison of meshes between vMAP [35] and the variants of our method on objects from the Replica dataset [75] after only 50 frames of optimization. Meshes are extracted with a resolution of 5mm and our models using object priors rely on ground truth retrieval and registration. While our reconstructions from scratch are more accurate than vMAP, adding object priors helps recovering faster the geometry and, optionally, texture of an object.

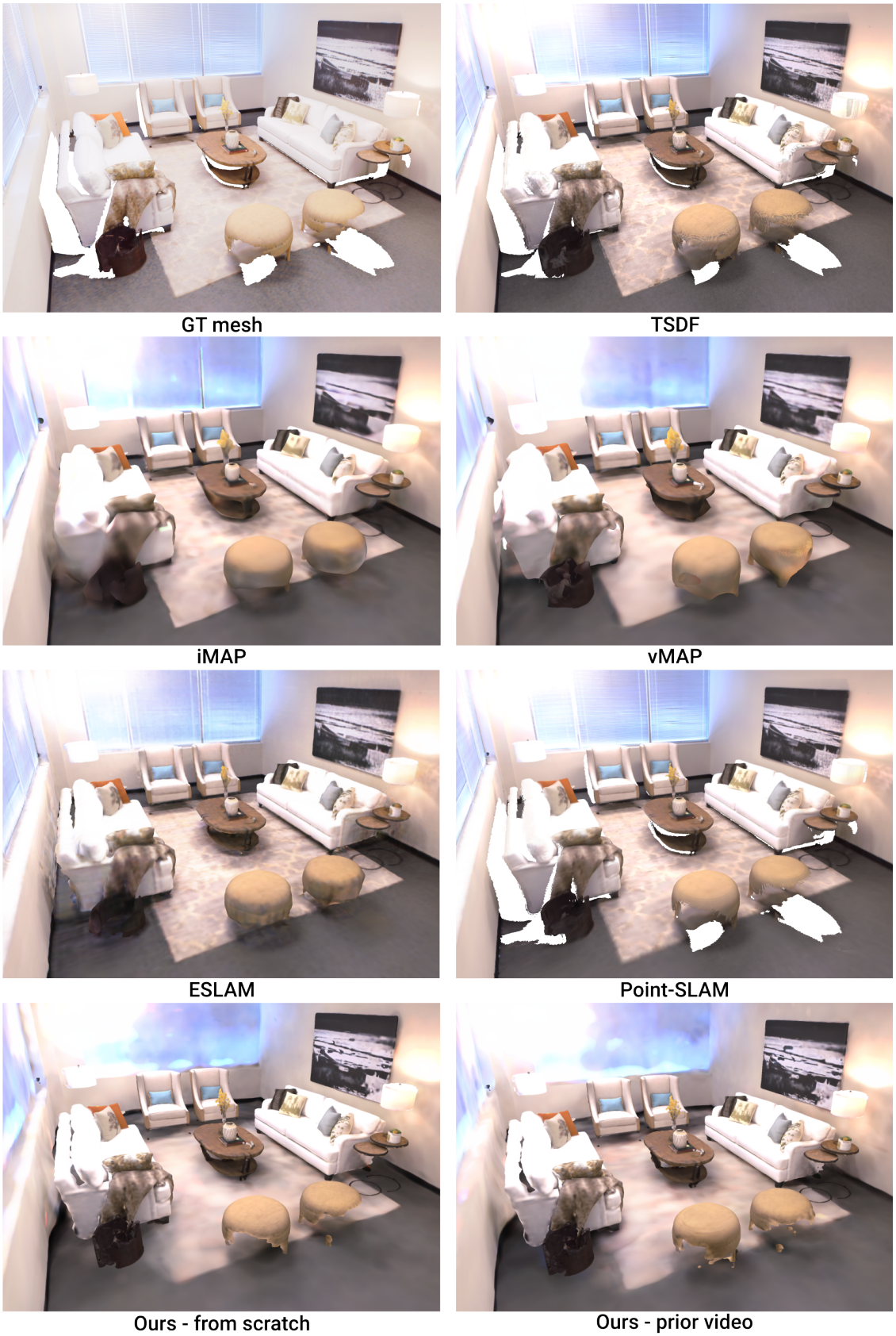

Figure 12. Visualization of reconstructed meshes at the scene level using TSDF-Fusion [10], iMAP [77] (through its reimplementation in [35]), vMAP [35], ESLAM [30], Point-SLAM [71] and our method, with or without object prior, on scene *room-0* of the Replica dataset [75].

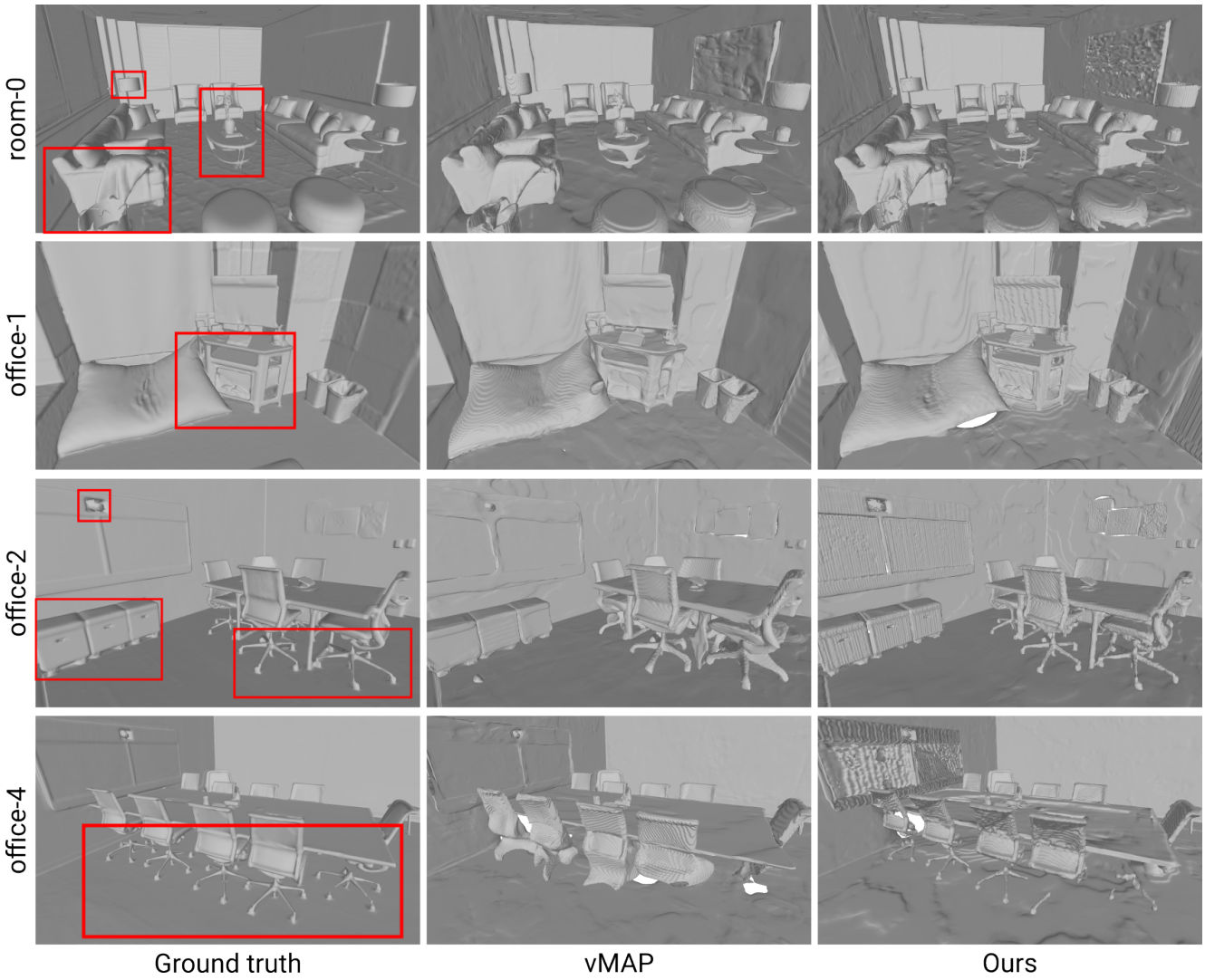

Figure 13. Textureless meshes reconstructed with vMAP [35] and our method on scenes of the Replica dataset [75]. The textured version is presented in Figure 4 of the main paper.

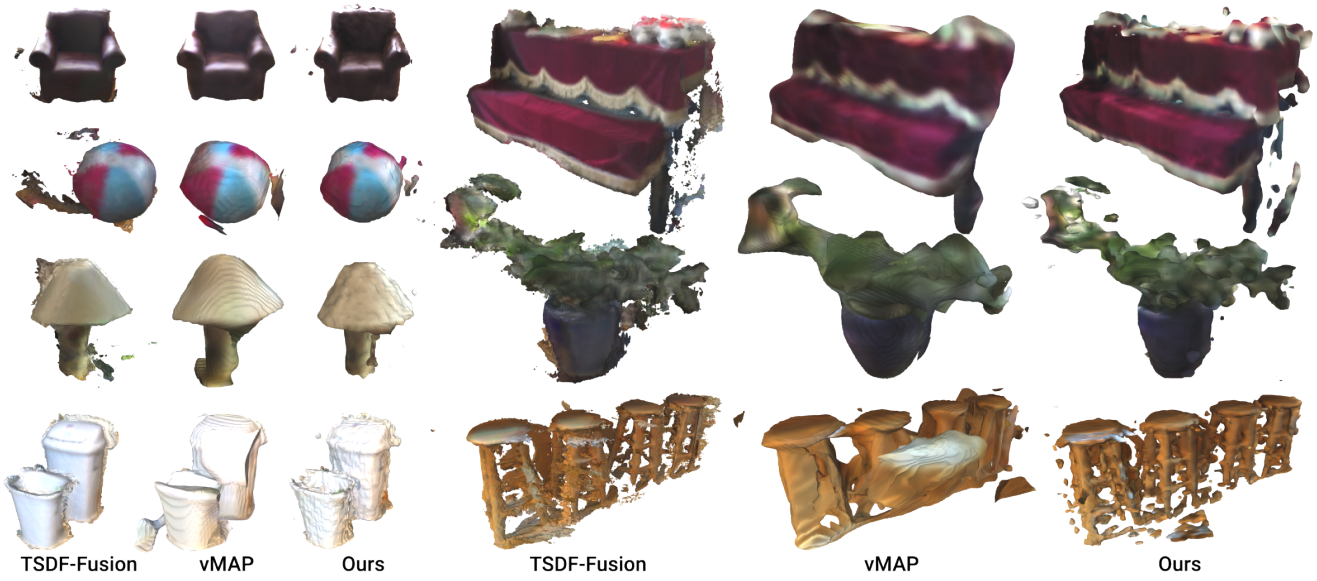

Figure 14. Comparison of meshes between TSDF [10, 95], vMAP [35] and our method on objects from the ScanNet dataset [12]. Meshes are extracted with a resolution of 1cm. Our reconstructions are more accurate than vMAP while being more sensitive about depth and segmentation errors than TSDF.

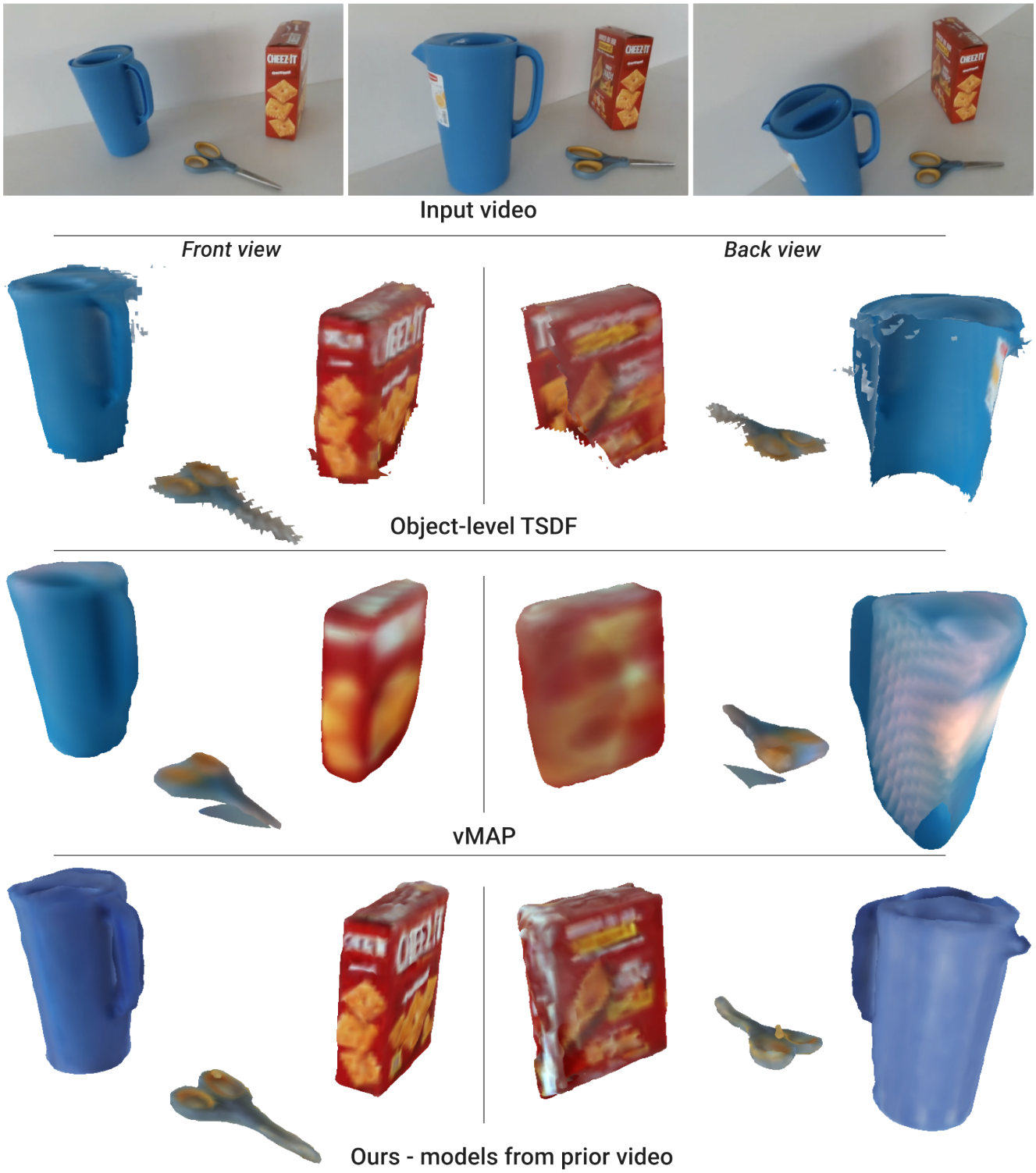

Figure 15. Additional reconstruction of a self-captured sequence with object-level TSDF, vMAP and our method using models from ground truth meshes, viewed from the front and the back.

Figure 16. Examples of rendered images of ground truth meshes using the Blenderproc [15] renderer. Our object models that leverage prior knowledge of GT meshes are fitted on these images.

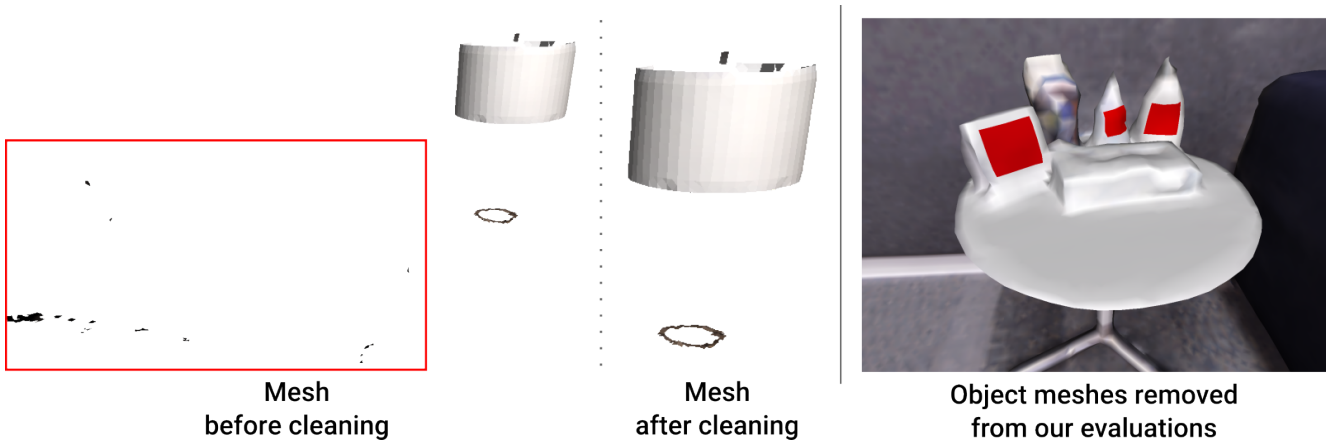

Figure 17. Examples of meshes cleaned before running our evaluations in the main paper. *(Left)*: few objects are cleaned to remove outliers, highlighted in red here. *(Right)*: other small and flat objects like product tags, colored in red, are also discarded.