vMAP: Vectorised Object Mapping for Neural Field SLAM

Xin Kong   Shikun Liu   Marwan Taher   Andrew J. Davison

Dyson Robotics Lab, Imperial College London

{x.kong21, shikun.liu17, m.taher, a.davison}@imperial.ac.uk

Figure 1. vMAP automatically builds an object-level scene model from a real-time RGB-D input stream. Each object is represented by a separate MLP neural field model, all optimised in parallel via vectorised training. We use no 3D shape priors, but the MLP representation encourages object reconstruction to be watertight and complete, even when objects are partially observed or are heavily occluded in the input images. See for instance the separate reconstructions of the armchairs, sofas and cushions, which were mutually occluding each other, in this example from Replica.

Abstract

We present vMAP, an object-level dense SLAM system using neural field representations. Each object is represented by a small MLP, enabling efficient, watertight object modelling without the need for 3D priors.

As an RGB-D camera browses a scene with no prior information, vMAP detects object instances on-the-fly, and dynamically adds them to its map. Specifically, thanks to the power of vectorised training, vMAP can optimise as many as 50 individual objects in a single scene, with an extremely efficient training speed of 5Hz map update. We experimentally demonstrate significantly improved scene-level and object-level reconstruction quality compared to prior neural field SLAM systems. Project page: https://kxhit.github.io/vMAP.

1. Introduction

For robotics and other interactive vision applications, an object-level model is arguably semantically optimal, with scene entities represented in a separated, composable way, but also efficiently focusing resources on what is important in an environment.

The key question in building an object-level mapping system is what level of prior information is known about the objects in a scene in order to segment, classify and re-construct them. If no 3D object priors are available, then usually only the directly observed parts of objects can be reconstructed, leading to holes and missing parts [4, 46]. Prior object information such as CAD models or category-level shape space models enable full object shape estimation from partial views, but only for the subset of objects in a scene for which these models are available.

In this paper, we present a new approach which applies to the case where no 3D priors are available but still often enables watertight object reconstruction in realistic real-time scene scanning. Our system, vMAP, builds on the attractive properties shown by neural fields as a real-time scene representation [31], with efficient and complete representation of shape, but now reconstructs a separate tiny MLP model of each object. The key technical contribution of our work is to show that a large number of separate MLP object models can be simultaneously and efficiently optimised on a single GPU during live operation via vectorised training.

We show that we can achieve much more accurate and complete scene reconstruction by separately modelling objects, compared with using a similar number of weights in a single neural field model of the whole scene. Our real-time system is highly efficient in terms of both computation and memory, and we show that scenes with up to 50 objects can be mapped with 40KB per object of learned parameters across the multiple, independent object networks.

We also demonstrate the flexibility of our disentangled object representation to enable recomposition of scenes with new object configurations. Extensive experiments have been conducted on both simulated and real-world datasets, showing state-of-the-art scene-level and object-level reconstruction performance.

2. Related Work

This work follows in long series of efforts to build real-time scene representations which are decomposed into explicit rigid objects, with the promise of flexible and efficient scene representation and even the possibility to represent changing scenes. Different systems assumed varying types of representation and levels of prior knowledge, from CAD models [28], via category-level shape models [10,11,32,36] to no prior shape knowledge, although in this case only the visible parts of objects could be reconstructed [15,27,38].

Neural Field SLAM Neural fields have recently been widely used as efficient, accurate and flexible representations of whole scenes [16, 17, 19, 22]. To adopt these representations into real-time SLAM systems, iMAP [31] demonstrated for the first time that a simple MLP network, incrementally trained with the aid of depth measurements from RGB-D sensors, can represent room-scaled 3D scenes in real-time. Some of iMAP's most interesting properties were its tendency to produce watertight reconstructions, even often plausibly completing the unobserved back of objects. These coherence properties of neural fields were particularly revealed when semantic output channels were added, as in SemanticNeRF [43] and iLabel [44], and were found to inherit the coherence. To make implicit representation more scalable and efficient, a group of implicit SLAM systems [25, 35, 40, 45, 48] fused neural fields with conventional volumetric representations.

Object Representations with Neural Fields However, obtaining individual object representations from these neural field methods is difficult, as the correspondences between network parameters and specific scene regions are complicated and difficult to determine. To tackle this, DeRF [23] decomposed a scene spatially and dedicated smaller networks to each decomposed part. Similarly, KiloNeRF [24] divided a scene into thousands of volumetric parts, each represented by a tiny MLP, and trained them in parallel with custom CUDA kernels to speed up NeRF. Different from KiloNeRF, vMAP decomposes the scene into objects which are semantically meaningful.

To represent multiple objects, ObjectNeRF [39] and ObjSDF [37] took pre-computed instance masks as additional input and conditioned object representation on learnable object activation code. But these methods are still trained offline and tangle object representations with the main scene network, so that they need to optimise the network weights with all object codes during training, and infer the whole network to get the shape of a desired object. This contrasts with vMAP which models objects individually, and is able to stop and resume training for any objects without any inter-object interference.

The recent work most similar to ours has used the attractive properties of neural field MLPs to represent single objects. The analysis in [5] explicitly evaluated the use of over-fit neural implicit networks as a 3D shape representation for graphics, considering that they should be taken seriously. The work in [1] furthered this analysis, showing how object representation was affected by different observation conditions, though using the hybrid Instant NGP rather than a single MLP representation, so it is not clear whether some object coherence properties would be lost. Finally, the CodeNeRF system [9] trained a NeRF conditioned on learnable object codes, again proving the attractive properties of neural fields to represent single objects.

We build on this work in our paper, but for the first time show that many individual neural field models making up a whole scene can be simultaneously trained within a real-time system, resulting in accurate and efficient representation of many-object scenes.

3. vMAP: An Efficient Object Mapping System with Vectorised Training

3.1. System Overview

We first introduce our detailed design for object-level mapping with efficient vectorised training (Section 3.2), and then explain our improved training strategies of pixel sampling and surface rendering (Section 3.3). Finally, we show how we may recompose and render a new scene with these learned object models (Section 3.4). An overview of our training and rendering pipeline is shown in Fig. 2.

3.2. Vectorised Object Level Mapping

Object Initialisation and Association To start with, each frame is associated with densely labelled object masks. These object masks are either directly provided in the dataset, or predicted with an off-the-shelf 2D instance segmentation network. Since those predicted object masks have no temporal consistency across different frames, we perform object association between the previous and the current live frame, based on two criteria: i) *Semantic Consistency*: the object in the current frame is predicted as the same semantic class from the previous frame, and ii) *Spatial Consistency*: the object in the current frame is spatially close to the object in the previous frames, measured by the mean IoU of their 3D object bounds. When these two criteria are satisfied, we assume they are the same object instance and represent them with the same object model. Otherwise, they are different object instances and we initialise
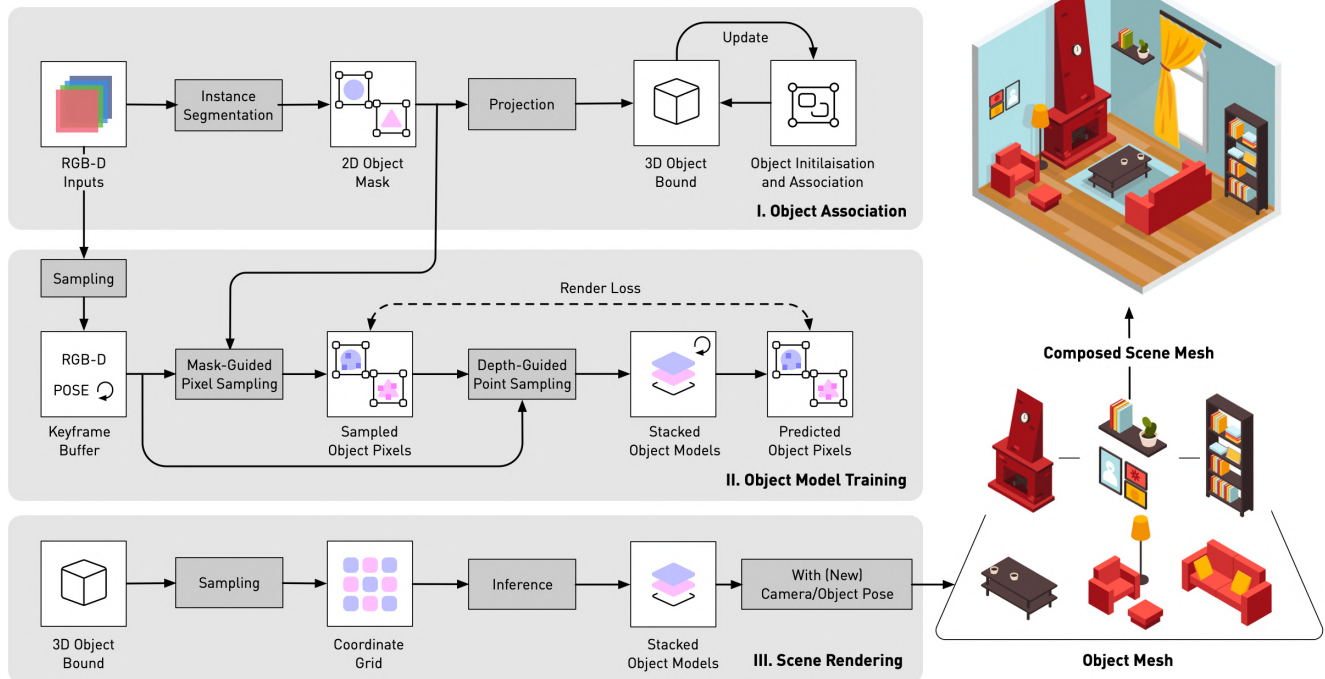
Figure 2. An overview of training and rendering pipeline of vMAP.

a new object model and append it to the models stack.

For each object in a frame, we estimate its 3D object bound by its 3D point cloud, parameterised by its depth map and the camera pose. Camera tracking is externally provided by an off-the-shelf tracking system, which we found to be more accurate and robust compared to jointly optimising pose and geometry. If we detect the same object instance in a new frame, we merge its 3D point cloud from the previous frames to the current frame and re-estimate its 3D object bound. Therefore, these object bounds are dynamically updated and refined with more observations.

Object Supervision We apply object-level supervision only for pixels inside a 2D object bounding box, for maximal training efficiency. For those pixels within an object mask, we encourage the object radiance field to be occupied and supervise them with depth and colour loss. Otherwise we encourage the object radiance field to be empty.

Each object instance samples training pixels from its own independent keyframe buffer. Therefore, we have flexibility to stop or resume the training of any object, with no training interference between objects.

Vectorised Training Representing a neural field with multiple small networks can lead to efficient training, as shown in prior work [24]. In vMAP, all object models are of the same design, except for the background object which we represent with a slightly larger network. Therefore, we are able to stack these small object models together for vec-

torised training, leveraging the highly optimised vectorised operations in PyTorch [8]. Since multiple object models are batched and trained simultaneously as opposed to sequentially, we optimise the use of the available GPU resources. We show that vectorised training is an essential design element to the system, resulting in significantly improved training speed, further discussed in Section 4.3.

3.3. Neural Implicit Mapping

Depth Guided Sampling Neural fields trained on RGB data only have no guarantee to model accurate object geometry, due to the fact that they are optimising for appearance rather than the geometry. To obtain more geometrically accurate object models, we benefit from the depth map available from an RGB-D sensor, providing a strong prior for learning the density field of 3D volumes. Specifically, we sample N_s and N_c points along each ray, for which N_s points are sampled with a Normal distribution centered around the surface t_s (from the depth map), with a small d_σ variance, and N_c points are uniformly sampled between the camera t_n (the near bound) and the surface t_s, with a stratified sampling approach. When the depth measurement is invalid, the surface t_s is then replaced with the far bound t_f. Mathematically, we have:

$$t_i \sim \mathcal{U}\left(t_n + \frac{i-1}{N_c}\left(t_s - t_n\right), t_n + \frac{i}{N_c}\left(t_s - t_n\right)\right), \quad (1)$$

$$t_i \sim \mathcal{N}(t_s, d_\sigma^2). \quad (2)$$

We choose $d_\sigma = 3cm$ which works well in our implementation. We observe that training more points near the surface helps to guide the object models to quickly focus on representing accurate object geometry.

Surface and Volume Rendering As we are concerned more by 3D surface reconstruction than 2D rendering, we omit the viewing direction from the network input, and model object visibility with a binary indicator (no transparent objects). With similar motivation to UniSURF [21], we parameterise the occupancy probability of a 3D point x_i as $o_\theta(x_i) \to [0, 1]$, where o_θ is a continuous occupancy field. Therefore, the termination probability at point x_i along ray $\mathbf{r}$ becomes $T_i = o(x_i) \prod_{j<i}(1 - o(x_j))$, indicating that no occupied samples x_j with $j < i$ exist before x_i. The corresponding rendered occupancy, depth and colour are defined as follows:

$$\hat{O}(\mathbf{r}) = \sum_{i=1}^{N} T_i, \ \hat{D}(\mathbf{r}) = \sum_{i=1}^{N} T_i d_i, \ \hat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i c_i. \quad (3)$$

Training Objective For each object k, we only sample training pixels inside that object's 2D bounding box, denoted by $\mathcal{R}^k$, and only optimise depth and colour for pixels inside its 2D object mask, denoted by M^k. Note that it is always true that $M^k \subset \mathcal{R}^k$. The depth, colour and occupancy loss for the object k are defined as follows:

$$L_{depth}^k = M^k \odot \sum_{\mathbf{r} \in R^k} |\hat{D}(\mathbf{r}) - D(\mathbf{r})|, \quad (4)$$

$$L_{colour}^k = M^k \odot \sum_{\mathbf{r} \in R^k} |\hat{C}(\mathbf{r}) - C(\mathbf{r})|, \quad (5)$$

$$L_{occupancy}^k = \sum_{\mathbf{r} \in R^k} |\hat{O}(\mathbf{r}) - M^k(\mathbf{r})|. \quad (6)$$

The overall training objective then accumulates losses for all K objects:

$$L = \sum_{k=1}^{K} L_{depth}^k + \lambda_1 \cdot L_{colour}^k + \lambda_2 \cdot L_{occupancy}^k. \quad (7)$$

We choose loss weightings $\lambda_1 = 5$ and $\lambda_2 = 10$, which we found to work well in our experiments.

3.4. Compositional Scene Rendering

Since vMAP represents objects in a purely disentangled representation space, we can obtain each 3D object by querying within its estimated 3D object bounds and easily manipulate it. For 2D novel view synthesis, we use the Ray-Box Intersection algorithm [14] to calculate near and far bounds for each object, and then rank rendered depths along each ray to achieve occlusion-aware scene-level rendering. This disentangled representation also opens up other types

of fine-grained object-level manipulation, such as changing object shape or textures by conditioning on disentangled pre-trained feature fields [20, 42], which we consider as an interesting future direction.

4. Experiments

We have comprehensively evaluated vMAP on a range of different datasets, which include both simulated and real-world sequences, with and without ground-truth object masks and poses. For all datasets, we qualitatively compare our system to prior state-of-the-art SLAM frameworks on 2D and 3D scene-level and object-level rendering. We further quantitatively compare these systems in datasets where ground-truth meshes are available. Please see our attached supplementary material for more results.

4.1. Experimental Setup

Datasets We evaluated on Replica [29], ScanNet [3], and TUM RGB-D [6]. Each dataset contains sequences with different levels of quality in object masks, depth and pose measurements. Additionally, we also showed vMAP's performance in complex real-world with self-captured video sequences recorded by an Azure Kinect RGB-D camera. An overview of these datasets is shown in Tab. 1.

	Object Masks	Depth Quality	Pose Estimation
Replica	Perfect GT	Perfect GT	Perfect GT
ScanNet	Noisy	Noisy	Perfect GT
TUM RGB-D	Detic	Noisy	ORB-SLAM3
Our Recording	Detic	Noisy	ORB-SLAM3

Table 1. An overview of datasets we evaluated.

Datasets with perfect ground-truth information represent the upper-bound performance of our system. We expect vMAP's performance in the real-world setting can be further improved, when coupled with a better instance segmentation and pose estimation framework.

Implementation Details We conduct all experiments on a desktop PC with a 3.60 GHz i7-11700K CPU and a single Nvidia RTX 3090 GPU. We choose our instance segmentation detector to be Detic [47], pre-trained on an open-vocabulary LVIS dataset [7] which contains more than 1000 object classes. We choose our pose estimation framework to be ORB-SLAM3 [2], for its fast and accurate tracking performance. We continuously update the keyframe poses using the latest estimates from ORB-SLAM3.

We applied the same set of hyper-parameters for all datasets. Both our object and background model use 4-layer MLPs, with each layer having hidden size 32 (object) and 128 (background). For object / background, we selected keyframes every 25 / 50 frames, 120 / 1200 rays each training step with 10 points per ray. The number of objects in a scene typically varies between 20 and 70, among which the

	TSDF-Fusion*	iMAP	iMAP*	NICE-SLAM	NICE-SLAM*	vMAP
Scene Acc. [cm] ↓	**1.28**	4.43	2.15	2.94	3.04	3.20
Scene Comp. [cm] ↓	5.61	5.56	2.88	4.02	3.84	**2.39**
Scene Comp. Ratio [<5cm %] ↑	82.67	79.06	90.85	86.73	86.52	**92.99**
Object Acc. [cm] ↓	**0.45**	-	3.57	-	3.91	2.23
Object Comp. [cm] ↓	3.69	-	2.38	-	3.27	**1.44**
Object Comp. Ratio [<5cm %] ↑	82.98	-	90.19	-	83.97	**94.55**
Object Comp. Ratio [<1cm %] ↑	61.70	-	47.79	-	37.79	**69.23**

Table 2. Averaged reconstruction results for 8 indoor Replica scenes. ∗ represents the baselines we re-trained with ground-truth pose.
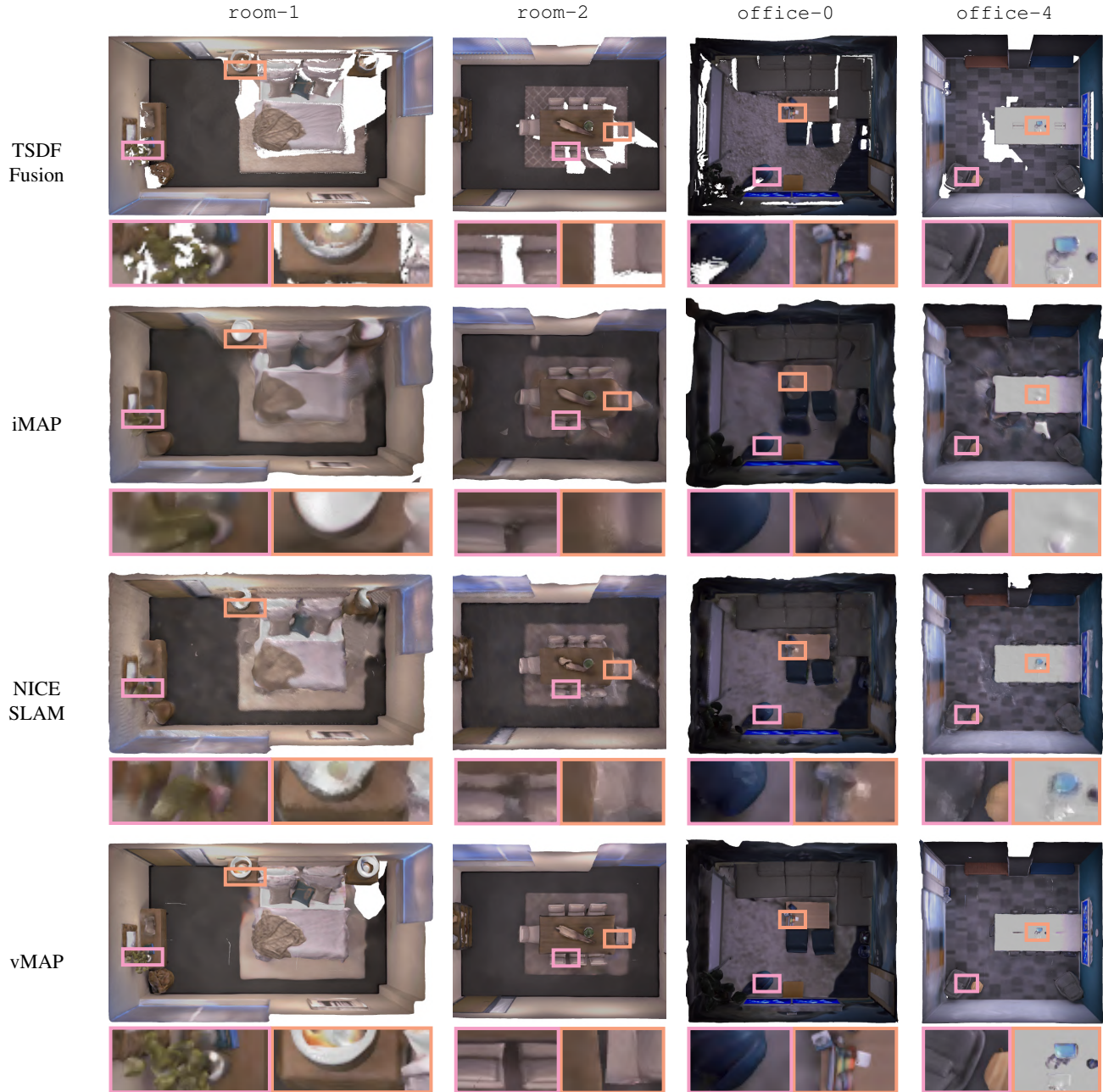

Figure 3. Scene reconstruction for 4 selected Replica scenes. Interesting regions are highlighted with coloured boxes, showing vMAP's significantly improved reconstruction quality. All scene meshes are provided by the original authors.

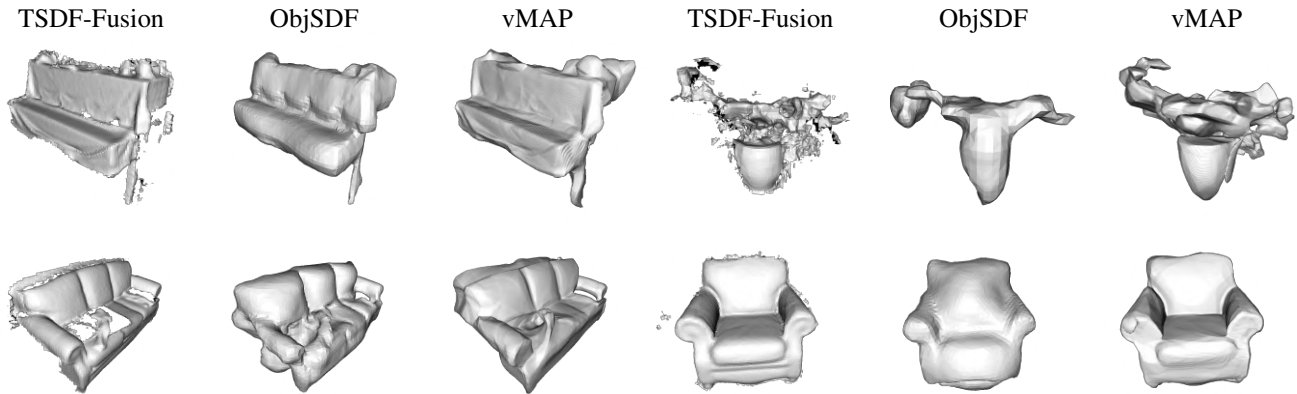| TSDF-Fusion | ObjSDF | vMAP | TSDF-Fusion | ObjSDF | vMAP |

Figure 4. Visualisation of object reconstructions with vMAP compared to TSDF-Fusion and ObjSDF. Note that all object reconstructions from ObjSDF require much longer off-line training. All object meshes from ObjSDF are provided by the original authors.

| NICE-SLAM* | vMAP |

Figure 5. Visualisation of scene reconstruction from NICE-SLAM* (left) and vMAP (right) in a selected ScanNet sequence. Interesting regions are zoomed in. NICE-SLAM* was re-trained with ground-truth poses.

largest number of objects are in Replica and ScanNet scenes with an average of 50 objects per scene.

Metrics Following the convention of prior work [31, 48], we adopt *Accuracy*, *Completion*, and *Completion Ratio* for 3D scene-level reconstruction metrics. Besides, we note that such scene-level metrics are heavily biased towards the reconstruction of large objects like walls and floors. Therefore, we additionally provide these metrics at the object-level, by averaging metrics for all objects in each scene.

4.2. Evaluation on Scene and Object Reconstruction

Results on Replica We experimented on 8 Replica scenes, using the rendered trajectories provided in [31], with 2000 RGB-D frames in each scene. Tab. 2 shows the averaged quantitative reconstruction results in these Replica indoor sequences. For scene-level reconstruction, we compared with TSDF-Fusion [46], iMAP [31] and NICE-SLAM [48]. To isolate reconstruction, we also provided results for these baselines re-trained with ground-truth pose (marked with *), with their open-sourced code for the fair comparison. Specifically, iMAP* was implemented as a special case of vMAP, when considering the entire scene

as one object instance. For object-level reconstruction, we compared baselines trained with ground-truth pose.

vMAP's significant advantage thanks to object-level representation is to reconstruct tiny objects and objects with fine-grained details. Noticeably, vMAP achieved more than 50 – 70% improvement over iMAP and NICE-SLAM for object-level completion. The scene reconstructions of 4 selected Replica sequences are shown in Fig. 3, with interesting regions highlighted in coloured boxes. The quantitative results for 2D novel view rendering are further provided in the supplementary material.

Results on ScanNet To evaluate on a more challenging setting, we experimented on ScanNet [3], a dataset composed of real scenes, with much noisier ground-truth depth maps and object masks. We choose a ScanNet sequence selected by ObjSDF [37], and we compared with TSDF-Fusion and ObjSDF for object-level reconstruction, and we compared with NICE-SLAM (re-trained with ground-truth pose) for scene-level reconstruction. Unlike ObjSDF, which was optimised from pre-selected posed images without depth for much longer off-line training, we ran both vMAP and TSDF-Fusion in an online setting with depth. As
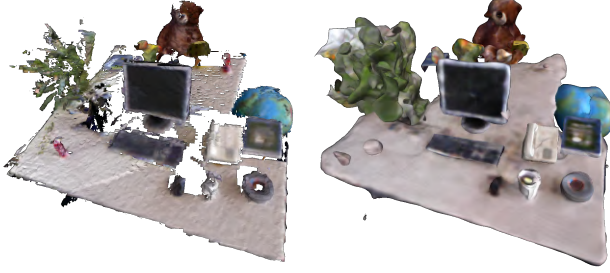
Figure 6. Visualisation of scene reconstruction from TSDF-Fusion (left) and vMAP (right) in a selected TUM RGB-D sequence, trained in real time for 99 seconds.

ATE RMSE [cm]↓	iMAP	NICE-SLAM	vMAP	ORB-SLAM2
fr1/desk	4.9	2.7	2.6	**1.6**
fr2/xyz	2.0	1.8	1.6	**0.4**
fr3/office	5.8	3.0	3.0	**1.0**

Table 3. Camera tracking results on TUM RGB-D.

shown in Fig. 4, we see that vMAP generates objects with more coherent geometry than TSDF-Fusion; and with much finer details than ObjSDF, though with a much shorter training time. And consistently, we can see that vMAP generates much sharper object boundaries and textures compared to NICE-SLAM, as shown in Fig. 5.

Results on TUM RGB-D We evaluated on a TUM RGB-D sequence captured in the real-world, with object masks predicted by an off-the-shelf pre-trained instance segmentation network [47], and poses estimated by ORB-SLAM3 [2]. Since our object detector has no spatio-temporal consistency, we found that the same object can be occasionally detected as two different instances, which leads to some reconstruction artifacts. For example, the object 'globe' shown in Fig. 6 was also detected as 'balloon' in some frames, resulting the 'splitting' artifacts in the final object reconstruction. Overall, vMAP still predicts more coherent reconstruction for most objects in a scene, with realistic hole-filling capabilities compared to TSDF-Fusion. However, we acknowledge that the completion of complete out-of-view regions (e.g., the back of a chair) is beyond the reach of our system due to the lack of general 3D prior.

Though our work focuses more on mapping performance than pose estimation, we also report *ATE RMSE* [30] in Tab. 3 following [31, 48], by jointly optimising camera pose with map. We can observe that vMAP achieves superior performance, due to the fact that reconstruction and tracking quality are typically highly interdependent. However, there is a noticeable performance gap compared to ORB-SLAM. As such, we directly choose ORB-SLAM as our external tracking system, which leads to faster training speed, cleaner implementation, and higher tracking quality.

Figure 7. Visualisation of table-top reconstruction (top) and individual object reconstructions (bottom), from vMAP running in real time using an Azure Kinect RGB-D camera for 170 seconds.

	NICE-SLAM*	iMAP	vMAP	vMAP (w/o BG)
Model Param. ↓	12.12M	0.32M	0.66M	0.56M
Runtime ↓	34min34s	12min29s	8min16s	6min01s
Mapping Time ↓	845ms	360ms	226ms	120ms

Table 4. vMAP is extremely memory-efficient and runs 1.5x and 4x faster than iMAP and NICE-SLAM respectively, with even higher performance gains without the background (BG) model.

Results on Live Kinect Data Finally, we show the reconstruction result of vMAP on a table-top scene, from running in real-time with an Azure Kinect RGB-D camera. As shown in Fig. 7, vMAP is able to generate a range of realistic, watertight object meshes from different categories.

4.3. Performance Analysis

In this section, we compare different training strategies and architectural design choices for our vMAP system. For simplicity, all experiments were done on the Replica Room-0 sequence, with our default training hyper-parameters.

Memory and Runtime We compared memory usage and runtime with iMAP and NICE-SLAM in Tab. 4 and Fig. 9, all trained with ground-truth pose, and with the default training hyper-parameters listed in each method, for fair comparison. Specifically, we reported the *Runtime* for training the entire sequence, and *Mapping Time* for training each single frame, given the exact same hardware. We can observe that vMAP is highly memory efficient with less than 1M parameters. We want to highlight that vMAP achieves better reconstruction quality, and runs significantly faster (∼5Hz) than iMAP and NICE-SLAM with 1.5x and 4x training speed improvement respectively.

Vectorised v.s. Sequential Training We ablated training speed with vectorised and sequential operations (for loops), conditioned on different numbers of objects and different
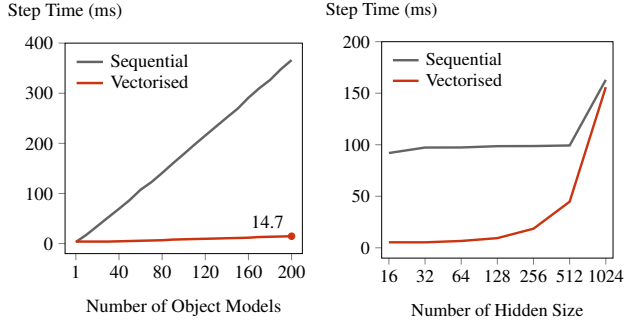
Figure 8. Vectorised operation allows extremely fast training speed compared to standard sequential operations using for loops.
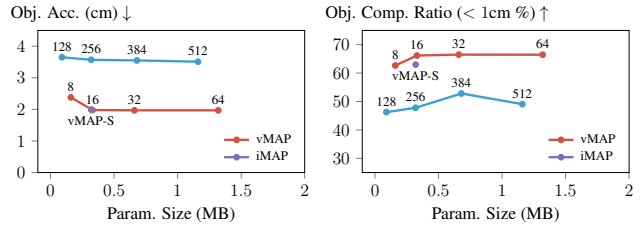

Figure 9. Object-level Reconstruction v.s. Model Param. (denoted by network hidden size). vMAP is more compact than iMAP, with the performance starting to saturate from hidden size 16.

sizes of object model. In Fig. 8, we can see that vectorised training enables tremendous improvements in optimisation speed, especially when we have a large number of objects. And with vectorised training, each optimisation step takes no more than 15ms even when we train as many as 200 objects. Additionally, vectorised training is also stable across a wide range of model sizes, suggesting that we can train our object models with an even larger size if required, with minimal additional training time. As expected, vectorised training and for loops will eventually have similar training speed, when we reach the hardware's memory limit.

To train multiple models in parallel, an initial approach we tried was spawning a process per object. However, we were only able to spawn a very limited number of processes, due to the per process CUDA memory overhead, which significantly limited the number of objects.

Object Model Capacity As vectorised training has minimal effect on training speed in terms of object model design, we also investigated how the object-level reconstruction quality is affected by different object model sizes. We experimented with different object model sizes by varying the hidden size of each MLP layer. In Fig. 9, we can see that the object-level performance starts to saturate starting from hidden size 16, with minimal or no improvement by further increasing model sizes. This indicates that object-level representation is highly compressible, and can be efficiently and accurately parameterised by very few parameters.

Stacked MLPs v.s. Shared MLP Apart from representing each object by a single individual MLP, we also explored a shared MLP design by considering multi-object mapping as a multi-task learning problem [26, 33]. Here, each object is additionally associated with a learnable latent code, and this latent code is considered as an conditional input to the network, jointly optimised with the network weights. Though we have tried multiple multi-task learning architectures [12, 18], early experiments (denoted as vMAP-S in Fig. 9) showed that this shared MLP design achieved slightly degraded reconstruction quality and had no distinct training speed improvement compared to stacked

MLPs, particularly when powered by vectorised training. Furthermore, we found that shared MLP design can lead to undesired training properties: i) The shared MLP needs to be optimised along with the latent codes from all the objects, since the network weights and all object codes are *entangled* in a shared representation space. ii) The shared MLP capacity is *fixed* during training, and therefore the representation space might not be sufficient with an increasing number of objects. This accentuates the advantages of disentangled object representation space, which is a crucial design element of vMAP system.

5. Conclusion

We have presented vMAP, a real-time object-level mapping system with simple and compact neural implicit representation. By decomposing the 3D scene into meaningful instances, represented by a batch of tiny separate MLPs, the system models the 3D scene in an efficient and flexible way, enabling scene re-composition, independent tracking and continually updating of objects of interest. In addition to more accurate and compact object-centric 3D reconstruction, our system is able to predict plausible watertight surfaces for each object, even under partial occlusion.

Limitations and Future Work Our current system relies on an off-the-shelf detector for instance masks, which are not necessarily spatio-temporally consistent. Though the ambiguity is partially alleviated by data association and multi-view supervision, a reasonable global constraints will be better. As objects are modelled independently, dynamic objects can be continually tracked and reconstructed to enable downstream tasks, e.g., robotic manipulation [34]. To extend our system to a monocular dense mapping system, depth estimation networks [13, 41] or more efficient neural rendering approaches [19] could be further integrated.

Acknowledgements

References

[1] Jad Abou-Chakra, Feras Dayoub, and Niko Sünderhauf. Implicit object mapping with noisy data. *arXiv preprint arXiv:2204.10516*, 2022. 2

[2] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics (T-RO)*, 2021. 4, 7

[3] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4, 6

[4] 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 Transactions on Graphics (ToG)*, 2017. 1

[5] Thomas Davies, Derek Nowrouzezahrai, and Alec Jacobson. On the effectiveness of weight-encoded neural implicit 3d shapes. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. 2

[6] Felix Endres, Jürgen Hess, Nikolas Engelhard, Jürgen Sturm, Daniel Cremers, and Wolfram Burgard. An Evaluation of the RGB-D SLAM System. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012. 4

[7] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4

[8] He Horace and Zou Richard. functorch: Jax-like composable function transforms for pytorch. https://github.com/pytorch/functorch, 2021. 3

[9] Wonbong Jang and Lourdes Agapito. Codenerf: Disentangled neural radiance fields for object categories. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 2

[10] Xin Kong, Xuemeng Yang, Guangyao Zhai, Xiangrui Zhao, Xianfang Zeng, Mengmeng Wang, Yong Liu, Wanlong Li, and Feng Wen. Semantic graph based place recognition for 3d point clouds. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2020. 2

[11] Guanglin Li, Yifeng Li, Zhichao Ye, Qihang Zhang, Tao Kong, Zhaopeng Cui, and Guofeng Zhang. Generative category-level shape and pose estimation with semantic primitives. In *Conference on Robot Learning (CoRL)*, 2022. 2

[12] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 8

[13] Xiaoyang Lyu, Liang Liu, Mengmeng Wang, Xin Kong, Lina Liu, Yong Liu, Xinxin Chen, and Yi Yuan. Hr-depth: High resolution self-supervised monocular depth estimation. In

[14] Alexander Majercik, Cyril Crassin, Peter Shirley, and Morgan McGuire. A ray-box intersection algorithm and efficient dynamic voxel rendering. *Journal of Computer Graphics Techniques (JCGT)*, 2018. 4

[15] John McCormac, Ronald Clark, Michael Bloesch, Andrew Davison, and Stefan Leutenegger. Fusion++: Volumetric object-level slam. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2018. 2

[16] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[17] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2

[18] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 8

[19] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 2022. 2, 8

[20] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 4

[21] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 4

[22] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[23] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. Derf: Decomposed radiance fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[24] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 2, 3

[25] Antoni Rosinol, John J Leonard, and Luca Carlone. Nerf-slam: Real-time dense monocular slam with neural radiance fields. *arXiv preprint arXiv:2210.13641*, 2022. 2

[26] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017. 8

[27] Martin Rünz and Lourdes Agapito. Co-fusion: Real-time segmentation, tracking and fusion of multiple objects.

Proceedings of the National Conference on Artificial Intelligence (AAAI), 2021. 8

In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017. 2

[28] Renato F Salas-Moreno, Richard A Newcombe, Hauke Strasdat, Paul HJ Kelly, and Andrew J Davison. SLAM++: Simultaneous Localisation and Mapping at the Level of Objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 2

[29] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 4

[30] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A Benchmark for the Evaluation of RGB-D SLAM Systems. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2012. 7

[31] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J. Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 1, 2, 6, 7

[32] Edgar Sucar, Kentaro Wada, and Andrew Davison. NodeSLAM: Neural object descriptors for multi-view shape reconstruction. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2020. 2

[33] Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2021. 8

[34] Kentaro Wada, Edgar Sucar, Stephen James, Daniel Lenton, and Andrew J Davison. Morefusion: Multi-object reasoning for 6d pose estimation from volumetric fusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 8

[35] Jingwen Wang, Tymoteusz Bleja, and Lourdes Agapito. Go-surf: Neural feature grid optimization for fast, high-fidelity rgb-d surface reconstruction. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2022. 2

[36] Jingwen Wang, Martin Rünz, and Lourdes Agapito. Dspslam: object oriented slam with deep shape priors. In *2021 International Conference on 3D Vision (3DV)*, 2021. 2

[37] Qianyi Wu, Xian Liu, Yuedong Chen, Kejie Li, Chuanxia Zheng, Jianfei Cai, and Jianmin Zheng. Object-compositional neural implicit surfaces. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 2, 6

[38] Binbin Xu, Wenbin Li, Dimos Tzoumanikas, Michael Bloesch, Andrew Davison, and Stefan Leutenegger. MID-Fusion: Octree-based object-level multi-instance dynamic slam. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2019. 2

[39] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 2

[40] Xingrui Yang, Hai Li, Hongjia Zhai, Yuhang Ming, Yuqian Liu, and Guofeng Zhang. Vox-Fusion: Dense tracking and mapping with voxel-based neural implicit representation. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2022. 2

[41] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 8

[42] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. Nerf-editing: geometry editing of neural radiance fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 4

[43] Shuaifeng Zhi, Michael Bloesch, Stefan Leutenegger, and Andrew J Davison. SceneCode: Monocular dense semantic reconstruction using learned encoded scene representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[44] Shuaifeng Zhi, Edgar Sucar, Andre Mouton, Iain Haughton, Tristan Laidlow, and Andrew J Davison. ilabel: Revealing objects in neural fields. *IEEE Robotics and Automation Letters (RA-L)*, 2022. 2

[45] Xingguang Zhong, Yue Pan, Jens Behley, and Cyrill Stachniss. Shine-mapping: Large-scale 3d mapping using sparse hierarchical implicit neural representations. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2023. 2

[46] Qian-Yi Zhou and Vladlen Koltun. Dense scene reconstruction with points of interest. *ACM Transactions on Graphics (ToG)*, 2013. 1, 6

[47] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 4, 7

[48] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 6, 7