

# Learning to Complete Object Shapes for Object-level Mapping in Dynamic Scenes

Binbin Xu<sup>1</sup>, Andrew J. Davison<sup>1</sup>, Stefan Leutenegger<sup>1,2</sup>

**Abstract**—In this paper, we propose a novel object-level mapping system that can simultaneously segment, track, and reconstruct objects in dynamic scenes. It can further predict and complete their full geometries by conditioning on reconstructions from depth inputs and a category-level shape prior with the aim that completed object geometry leads to better object reconstruction and tracking accuracy. For each incoming RGB-D frame, we perform instance segmentation to detect objects and build data associations between the detection and the existing object maps. A new object map will be created for each unmatched detection. For each matched object, we jointly optimise its pose and latent geometry representations using geometric residual and differential rendering residual towards its shape prior and completed geometry. Our approach shows better tracking and reconstruction performance compared to methods using traditional volumetric mapping or learned shape prior approaches. We evaluate its effectiveness by quantitatively and qualitatively testing it in both synthetic and real-world sequences.

## I. INTRODUCTION

Simultaneous Localisation and Mapping (SLAM) research aims to concurrently estimate both the scene geometry of the unknown environment as well as the robot pose within it from the data of its on-board sensors only. It has rapidly progressed from sparse SLAM [1], [2] into dense SLAM [3], [4], and recently into semantic object-level SLAM [5], [6]. This fast-evolving research has enabled many robotic applications. Despite this, most SLAM research still assumes a static scene, where points in the 3D world maintain constant spatial positions in a global coordinate. Any information violating this assumption, such as moving objects in the environment, would be treated as outliers and are intentionally ignored in tracking and mapping steps.

This setup, however, can only handle a small amount of dynamic elements, excluding itself from many real-world applications as environments, particularly where humans are involved, are continually changing. A robust SLAM system capable of handling highly dynamic environments, therefore, is desirable. Most current dynamic SLAM research can be categorised into three main directions. One maps the whole changing world in a non-rigid deformable representation to deal with the changing topology of deformable/moving objects [7]. The second aims at improving the robustness and accuracy of camera tracking by ignoring all possibly

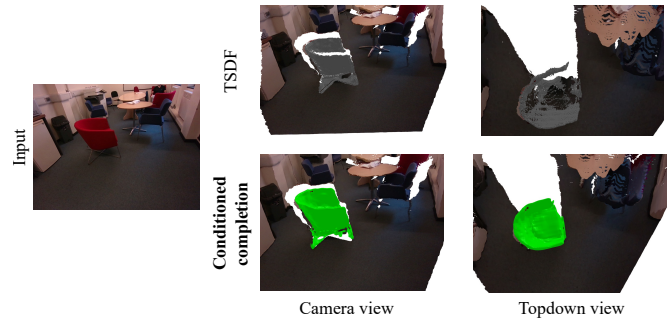


Fig. 1. Given RGB-D images, our system builds object-level dense dynamic maps that can robustly track camera pose and object poses while completing the missing sensor information using object priors. Compared to the classic TSDF maps, our object maps fill in unobserved parts and their latent codes can be optimised jointly with object poses. Interfered regions by humans can be detected and intentionally removed in the system. The background pointclouds are projected for pure visualisation purpose.

moving objects and building a single static background model [8], [9], [10]. The third models dynamic environments by creating object-centric maps for each possibly moving rigid object in the scene while fusing corresponding information into these object-level maps [11], [12]. Object-level tracking and mapping can be conducted for each object and camera motion against the static part of the map. This paper aligns with the last direction as we believe that, similar to human perception, an instance-awareness of the surrounding environment can help intelligent robots perceive the scene changes and enables meaningful interactions with the surrounding environment.

By far most existing object-level dynamic SLAM systems mentioned above adopt the classic map representation that have been exploited in the static SLAM systems, such as pointclouds [13], surfels [11] or volumetric maps [12]. This leads to partial or incomplete object maps as only the observable information can be fused into the object models. Information in unseen parts can not be filled unless an object or the sensor is moved actively. Contrary to reconstructing objects from scratch, some works recently explored learning-based category-level object shape priors and build object-level maps based on learned shape priors [14], [15]. The object geometry and pose are usually optimised via differentiable rendering. However, most of these systems are only applicable in static scenes. Besides, despite being able to generate complete object geometry, object shape priors cannot capture complex geometry details as the bottleneck of its latent representation can only interpolate shapes inside the training datasets [16]. When combined with dense

<sup>1</sup> The authors are with Department of Computing, Imperial College London, United Kingdom. {b.xu17, a.davison, s.leutenegger}@imperial.ac.uk

<sup>2</sup> The author is also with the Smart Robotics Lab, Technical University of Munich, Germany {stefan.leutenegger}@tum.de

The supplementary video can be watched on: <https://youtu.be/mH22H7jp1D8>

image alignment, such as photometric or ICP residuals, this inconsistency between the measurement and the object prior model inevitably leads to inaccurate object motion trajectory estimates.

This work stands in the middle between reconstructing object geometry from scratch and mapping using object shape priors. We reconstruct the observable part by continuously fusing depth measurements into a volumetric canonical space and predict the complete geometry by conditioning it on the fused volume. The resulting object geometry can preserve the details that have been observed in the past and simultaneously complete the missing geometry information. We also verified that this completed object geometry can further improve the accuracy of object tracking. The main contributions in this paper can be summarised as follows:

- 1) we present, to the best of our knowledge, the first RGB-D object-level dynamic mapping system that can complete unseen parts of objects using a shape prior encoded in neural networks while still reconstructing observed parts accurately;
- 2) a joint optimisation of object pose and shape geometry based on geometric residuals and differentiable rendering;
- 3) extensive experiments of object tracking and reconstruction components on synthetic and real-world data to evaluate the benefits of object geometry completion for object-level SLAM.

## II. RELATED WORKS

*a) Object-level dynamic SLAM:* Although object-level dynamic SLAM research can be dated back to [17], visual dense object-level dynamic SLAM has only been explored recently. From RGB-D sensor inputs, Co-Fusion [18] segments objects by either ICP motion segmentation or semantic segmentation and then tracks objects separately based on ElasticFusion [19]. MaskFusion [11] segments objects using a combination of instance segmentation from Mask-RCNN and geometric edges, and tracks objects using the same approach as Co-Fusion. Both Co-Fusion and MaskFusion use surfels to represent map models, which is memory efficient but cannot directly provide free space information in the map, and neither surface connectivity. DynSLAM-II [13] extends from ORB-SLAM II [20] and formulates object tracking using sparse feature descriptor matching. Object maps are represented using clusters of pointclouds, which can bring object poses and geometries into the pose graph optimisation but also lack space connection awareness. Instead, MID-Fusion [12] uses memory-efficient octree-based volumetric signed distance field (SDF) representation for objects and re-parametrises tracking residuals in object coordinates and weights. EM-Fusion [21] similarly uses volumetric SDF object maps but formulates object tracking as direct alignment of input frames with the SDF representations. Their following work [22] infers the missing object geometry by penalising the hull and intersection constraints. However, it did not explore shape prior information and requires heavy computation to optimise SDF field explicitly. Instead, we

fuse the depth measurement into object-level SDF maps and predict completed object geometries by incorporating a shape prior in continuous occupancy fields.

*b) SLAM with shape prior maps:* Instead of estimating both object geometry and poses from scratch, some approaches use a shape prior to represent objects. Since the coordinates of object shape priors and the run-time measurement are not necessarily aligned, a relative rigid transformation needs to be estimated. This is analogous to the localisation-only problem in SLAM. SLAM++ [23] is one of the pioneering object-level RGB-D SLAM systems. It scanned objects in advance and then maps the detected instances at run-time by jointly optimising a pose graph of camera and object poses. Relying on pre-scanned objects, however, limits its ability to scale to unknown object models. Rather than using instance-level shape priors, several following works exploited category-level shape priors as there is limited variance in certain object categories. The category-level shape prior can be learnt in various representations, such as PCA models as in DirectShape [24], occupancy grids as in Deep-SLAM++ [25], variational autoencoders as in NodeSLAM [14], or autoencoders as in DSP-SLAM [15]. However, most of these works only target static environments, as multi-view consistency of static world points is required to localise the shape prior models. [26] relax this restriction using a Bayesian filter to associate object detections on different frames and fuse the prior model by simply averaging the latent codes from each frame. However, object shape deviations cannot always be captured by the shape prior interpolation. The object tracking accuracy would be affected by the discrepancy between the prior shape model and the online measurement. We address this challenge by conditioning the completion network on the integrated 3D reconstruction model.

*c) Object-level tracking:* To track moving objects in RGB-D sequences, several pioneering object-level works adopt the frame-to-model tracking methods from RGB-D SLAM systems [18] and parametrise them for object tracking [12], [13]. The classic photometric residual, however, is difficult to deal with as object lighting changes; several approaches explore using learning-based robust features to formulate object tracking in direct [27] and in-direct ways [28]. Parallel to learning category-level shape priors, Wang et al. [29] proposed to learn category-level pixel-wise correspondence from RGB-D images to the canonical space. The shape is implicitly defined from this correspondence and the frame-to-canonical transformation can be estimated from this noisy correspondence. Rempe et al. [30] further proposed to generate more stable correspondences by accumulating temporal information from RGB-D sequences. Recently, Muller et al. [31] proposed to track moving objects and predict their complete geometry using such canonical correspondence representation. In this work, the object pose is initially predicted using canonical correspondence regression. However, we found it does not necessarily yield alignment to the canonical space and we further optimise the pose tracking using geometric residual and differential

rendering.

### III. METHOD

#### A. Notations and Preliminaries

In this work, we will use the following notation: a coordinate frame is denoted as  $\mathcal{F}_A$ . The homogeneous transformation from  $\mathcal{F}_B$  to  $\mathcal{F}_A$  is denoted as  $T_{AB}$ , which is composed of a rotation matrix  $C_{AB}$  and a translation vector  ${}_A\mathbf{r}_{AB}$ .

Every detected object is represented in its individual object coordinate frame  $\mathcal{F}_{O_n}$ , with  $n \in \{0 \dots, N\}$ , where  $N$  is the total number of objects (excluding background) and 0 denotes background. We assume a canonical static volumetric model is stored in each object coordinate frame, forming the basis of our multi-instance SLAM system. To leverage the category-level shape prior, we need to align the canonical space with the one defined in training, otherwise the completion performance will be deteriorated since it cannot fully take advantage of the shape variances of the objects in the same category. The relative transformation between the current world coordinate and the corresponding object canonical space is defined as a joint state composed of a rigid transformation  $T_{WO_n}$  and the object scale  $s_{O_n}$ . We define the object pose using this joint state.  $T_{WO_n}$  needs to be continuously updated for a moving object but the object scale should be consistent across different frames.

#### B. System Overview

Figure 3 shows the pipeline of our proposed system. Each input RGB-D image is processed by Mask R-CNN to perform instance segmentation. The camera pose is tracked against background regions, excluding the human mask area and moving objects, similar to what has been proposed in MID-Fusion [12].

The object geometry is composed of two nodes with a shared object pose: prior node and posterior node, as shown in Figure 2. The prior node represents its category-level shape prior using a latent code  $\mathbf{z}_0 \in \mathbb{R}^{64}$ . It can be used to express the continuous SDF field  $s$  on any query 3D location in object canonical coordinate  $\mathbf{v} \in \mathbb{R}^3$  using a DeepSDF shape prior network  $F_0$  [16] as

$$s = F_0(\mathbf{v}, \mathbf{z}_0). \quad (1)$$

The prior node is used to initialise the object pose and re-localise the object model when object tracking is lost as its shape is not affected by measurements. The posterior node encodes a fused partial TSDf volume and its associated TSDf weight volume into TSDf feature volume  $\theta_t$  and TSDf confidence volume  $\theta_c$  separately using 3D-UNet [32]. Then a complete occupancy field  $o$  can be predicted on any given 3D position  $\mathbf{v} \in \mathbb{R}^3$  using a shape completion network  $F_1$ :

$$o = F_1(\mathbf{v}, \theta_t[\mathbf{v}], \theta_c[\mathbf{v}], \mathbf{z}_1). \quad (2)$$

where  $\theta_t[\mathbf{v}] \in \mathbb{R}^{32}$  and  $\theta_c[\mathbf{v}] \in \mathbb{R}^1$  denote the feature vectors tri-linearly interpolated at the point  $\mathbf{v}$  inside the volumes  $\theta_t$  and  $\theta_c$ , respectively. We additionally condition it on a latent code  $\mathbf{z}_1 \in \mathbb{R}^{32}$  so that the hidden space can be optimised

to generate novel shapes. The shape completion network  $F_1$  shares a similar architecture to the CONet [33], but also takes extra inputs of TSDf confidence weight and an instance-level latent code.

We perform an efficient Axis Aligned Bounding Box (AABB) ray intersection test [34] to find all the visible existing object models in the current viewpoint and render object masks for each visible models. An Intersection of Union (IoU) between the detections on the current frame and the rendered model masks is computed to build data associations between the current frame detections and existing object models. Then we track each object model and complete its geometry by performing a joint optimisation of pose and geometry (Section III-C). Using estimated poses of the camera and objects, new depth measurements are fused into an object model and a complete shape geometry can be predicted by conditioning on the fused model. New objects are created for unmatched detections by initialising their initial object pose using object prior models.

#### C. Joint Optimisation of Object Pose and Geometry

Instead of defining an arbitrary canonical space for object coordinates [12], we need to estimate the 7DoF relative transformation, which is composed of  $T_{WO_n}$  and  $s_{O_n}$ , to align the initialised object coordinate to the training canonical space for each object detection in order to take advantage of the learned prior information.

*a) Initialisation:* Given an RGB-D frame  $(I_L, D_L)$  and detected object mask  $M_n$ , we predict their positions in the canonical space  $\mathbf{v}$  and associated confidences  $w$  from back-projected pointcloud using a modified normalised object correspondence network  $F_n$  from [30]:

$$c_L \mathbf{v}(\mathbf{u}_L) = \pi^{-1}(\mathbf{u}_L, D_L[\mathbf{u}_L]), \forall \mathbf{u}_L \in M_n, \quad (3)$$

$$F_n(c_L \mathbf{v}) \rightarrow \mathbf{v}, w \quad (4)$$

Then we solve the 7-DoF relative transformation, scale  $s_{O_n}$ , rotation  $C_{C_L O_n}$ , and translation  ${}_{C_L}\mathbf{r}_{C_L O_n}$  from the regressed correspondences using the Umeyama algorithm [35] with SVD decomposition:

$$\operatorname{argmin}_{s_{O_n}, T_{C_L O_n}} \sum_{\mathbf{u}_L \in M_n} w \left( \mathbf{v} - \frac{1}{s_{O_n}} T_{C_L O_n}^{-1} c_L \mathbf{v}(\mathbf{u}_L) \right). \quad (5)$$

For the latent codes  $\mathbf{z}_0$  and  $\mathbf{z}_1$ , we use both zero code initialisations. We only run this pose initialisation step for new unmatched object detections. The initial pose solved from SVD decomposition, however, is necessary to be close to the ground-truth canonical pose, due to the unseen shapes or viewpoints, affecting the performance of shape completion.

*b) Coarse Estimation:* To refine object canonical poses from the initial pose prediction, we jointly optimise it with the prior latent code  $\mathbf{z}_0$  to minimise the 3D SDF loss  $E_{\text{SDF}}$  and 2D rendering loss  $E_{\text{render}}$ :

$$E_{\text{coarse}} = \lambda_s E_{\text{SDF}} + \lambda_{r0} E_{\text{render}} + \lambda_{z0} \|\mathbf{z}_0\|. \quad (6)$$

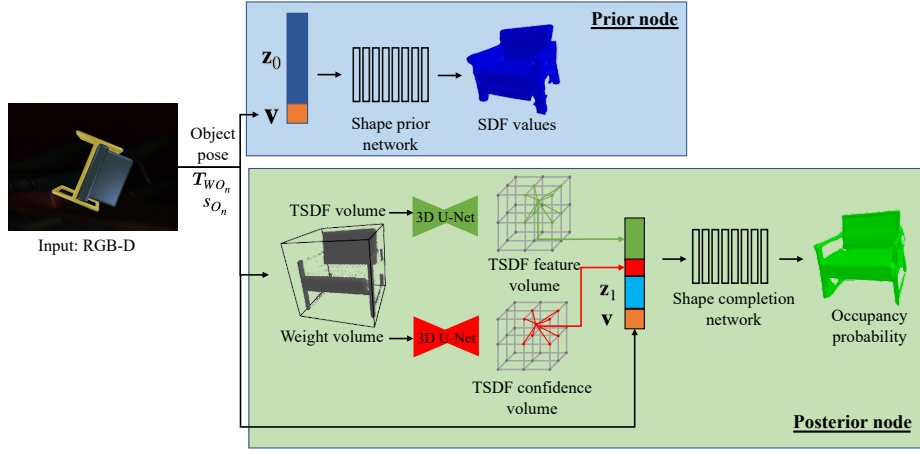


Fig. 2. The overview of our object geometry representation.

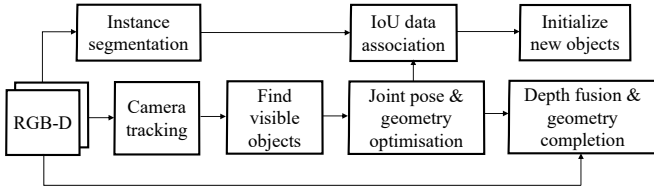


Fig. 3. The pipeline of the proposed method

The 3D SDF loss is defined to encourage the back-project depth points to align with the object surface, where the zero SDF value is defined

$$E_{\text{SDF}} = \sum_{\mathbf{u}_L \in M_n} F_0 \left( \frac{1}{s_{O_n}} \mathbf{T}_{C_L O_n}^{-1} C_L \mathbf{v}(\mathbf{u}_L), \mathbf{z}_0 \right). \quad (7)$$

We cannot compute SDF residuals for empty space since ground-truth SDF values are not available at test time. Instead, for the non-surface areas, we use differentiable rendering to encourage the rendered depth  $D_L$  to be close to the measured depth  $\tilde{D}_L$ . We compute the rendering loss for the visible 3D space inside the object 3D bounding box:

$$E_{\text{render}} = \sum_{\mathbf{u}_L \in B_n} D_L[\mathbf{u}_L] - \tilde{D}_L[\mathbf{u}_L], \quad (8)$$

where

$$D_L[\mathbf{u}_L] = \sum_{i=1}^N w_i d_i, \quad (9)$$

and  $w_i$  is the ray-termination probability [14] of sample  $i$  at depth  $d_i$  along the ray from the pixel  $\mathbf{u}_L$ ,

$$w_i = o_i \prod_{j=1}^{i-1} (1 - o_j), \quad (10)$$

and  $B_n$  is the 2D bounding box rendering from the estimated object 3D bounding box on this frame. A continuous occupancy field can be extracted from the continuous prior SDF field as proposed in [15]:

$$o_i = -\frac{1}{2\sigma} F_0 \left( \frac{1}{s_{O_n}} \mathbf{T}_{C_L O_n}^{-1} \left( \pi^{-1}(\mathbf{u}_L, d_i) \right), \mathbf{z}_0 \right), \quad (11)$$

where  $\sigma$  is the truncation distance to control the transition.

By freezing the network weight in  $F_n$ , the cost function in Equation (6) can be iteratively solved using Gauss-Newton optimisation with analytical Jacobians. Since the prior shape is not necessarily aligned with the actual observation, it is unnecessary to sample every pixel ray. Instead, we run this optimisation on sparse ray samples, which can speed up the optimisation without losing much accuracy.

*c) Dense Refinement:* After the coarse estimation, we have a rough alignment of the object model with the canonical space. However, the optimised object prior  $\mathbf{z}_0$  cannot necessarily capture the details of depth measurements. To further align the object model and to complete the hidden part, we jointly optimise the posterior latent code  $\mathbf{z}_1$  with the object pose by minimizing a 3D occupancy loss  $E_{\text{occ}}$  on both occupied and empty space (excluding the unknown 3D space) and a similar 2D rendering loss  $E_{\text{render}}$ :

$$E_{\text{refine}} = \lambda_o E_{\text{occ}} + \lambda_{r1} E_{\text{render}} + \lambda_{z1} \|\mathbf{z}_1\|. \quad (12)$$

The occupied space is defined on the back-projected points and the empty space is uniformly sampled in the background space as well as the foreground space before the depth measurement. The occupancy loss is defined using the binary cross-entropy loss between the predicted occupancy value  $o_v$  from the shape completion network using Equation (2) and the ground-truth occupancy values  $o_v^*$  (0.5 for the occupied space and 0 for the empty space) for sampled points  $\mathbf{v}$  inside the occupied and empty space:

$$E_{\text{occ}} = -\sum_{\mathbf{v}} [o_v \log(o_v^*) + (1 - o_v) \log(1 - o_v^*)]. \quad (13)$$

Similar to the coarse estimation, we can also evaluate the 2D rendering loss using Equation (8). The difference is here we use the decoded continuous occupancy value for the sampled  $d_i$  using Equation (2), instead of converting it from the SDF field in Equation (11). The refined object pose  $\mathbf{T}_{C_L O_n}$  is used to integrate the current depth frame into the corresponding TSDF volume and weight volume [3].

#### D. Training Setup

The learnable network parameters in this work includes three part, canonical correspondence network  $F_n$ , shape prior network  $F_0$ , shape posterior network  $F_1$ .

We train the canonical correspondence network using the partial pointcloud generated from the synthetic shapenet dataset [36]. During training, we augmented the input pointcloud with random object pose and solve the 7DoF object poses using Equation (5). To help network prediction robust to outliers, we also added random depth outliers in the pointcloud generation to learn the correspondence confidence  $w$  in a self-supervised way. The solved pose is compared to the augmented ground-truth pose and the whole network is trained end-to-end since the estimation is differentiable.

We use the pre-trained off-shelf network weights for the category-level shape prior network  $F_0$  [16], which was also trained in the shapenet dataset [36]. To train the posterior shape completion network, we rendered depth maps for each object in the shapenet dataset [36] and trained the shape completion network using the partial depth observation. We use the occupancy loss defined in Equation (13) to encourage the completed shape to be similar to the ground-truth one. Similar to the latent code training in DeepSDF [16], different object shapes have their own latent codes, which are trained together with the network. We make different partial observations of the same object shape share the same latent code.

### IV. EXPERIMENTS

#### A. Quantitative Reconstruction Evaluations

1) *Experimental Setup*: We validate the reconstruction quality of our method on object-level surface reconstruction tasks. We conduct a comparison on the chair category of the ShapeNet [36] dataset. The split of train/val/test sets follows the same setting in [33]. We randomly select 50 models from the test set to conduct quantitative evaluations. We generate input depth images by rendering images using uniformly sampled virtual camera viewpoints surrounding the CAD model. The hyperparameters used in inference optimization are chosen as  $\sigma = 0.05$ ,  $\lambda_s = 100$ ,  $\lambda_{r0} = 2.5$ ,  $\lambda_{z0} = 5$ ,  $\lambda_o = 100$ ,  $\lambda_{r1} = 1$ , and  $\lambda_{z1} = 1$ .

2) *Baseline Methods*: To evaluate the object mapping, we compare with the following baseline methods:

- TSDF-fusion: We fuse the depth measurements into a TSDF volume grid as in [3].
- DeepSDF mapping: We use the pre-trained decoder weight in [16]. As the shape completion code is not provided, we optimise the SDF loss on the input pointcloud as well as the empty space constraint proposed in IGR [37].
- CONet: We use the weights trained from partial pointcloud input in [33] and pass the accumulated pointcloud in the canonical space to generate the continuous occupancy field where the meshes are extracted.

3) *Metrics*: To quantitatively evaluate the quality and completeness of the shape reconstruction, we use the following metrics:

- IoU: We sample 100k points uniformly in the bounding box and evaluate on both the reconstructed and the ground-truth meshes whether each point is inside or outside. The final value is the fraction of intersection over union. Higher is better.
- Chamfer Distance: we sample 100k points on the surface of both the ground-truth and the reconstructed mesh. We compute the closest points from the reconstructed to the ground-truth mesh using kD-tree and vice-versa. We then compute the average of the L1 distances to the closest points in each direction. Lower is better.
- (In-)completeness: As the completeness of the object map is essential in this work, we also report completeness, which is the one-way chamfer distance from the ground-truth meshes to the reconstructed ones. This is to measure the closest distance from each ground-truth mesh points to the reconstruction. Lower is better.

4) *Results and Discussions*: We quantitatively evaluate how the view number of depth measurement would impact the reconstruction results of different methods. Figure 4 reports the result. It can be seen that our proposed method consistently show better reconstruction results from single view depth completion to multiple views. When the view number is limited, classic TSDF-Fusion shows worse result as it can only reconstruct the visible parts. CONet completes some missing information, but still struggles as it heavily depends on the input pointcloud. DeepSDF does not condition on the input and the latent code optimisation can fit the few depth measurement and shows better completion and reconstruction results. Our proposed method uses both the input information and shape prior information, yielding best performance. When more depth measurements are received, TSDF-Fusion and CONet start to fill in the missing information while DeepSDF struggles to leverage more measurement information. Our result also improves since we can also take advantage of the measurement information. Figure 5 shows an examples of reconstruction results by each method in the view number case of 1, 5, and 10.

#### B. Quantitative Tracking Evaluations

1) *Experimental Setup*: To quantitatively evaluate the object-level tracking and mapping performance, we randomly select 10 object models from the test split of the chair category in the ShapeNet [36] and render 200 frames using Blender. To ensure diversity of object motion, texture, and illuminations, we randomise four point light sources, camera viewpoint, and object trajectories. We then subsample the sequences using sampling intervals 1, 2, 4 in order to create small, medium and large motion subsets.

2) *Baseline Methods*: To evaluate the object tracking, we compare with the following baseline methods:

- RGB-D VO: a non-learning-based visual odometry method proposed in [38], which minimises the photo-

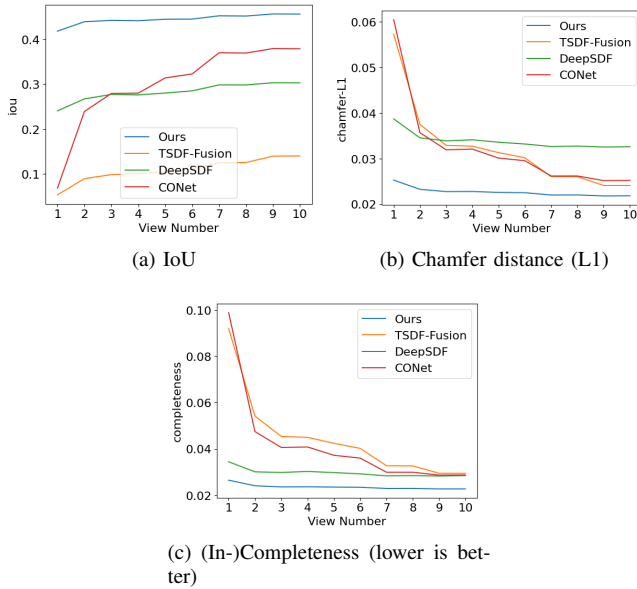


Fig. 4. Quantitative comparison of reconstruction quality and completion of our proposed methods v.s. classic TSDF-Fusion, learning-based DeepSDF and CONet. Our proposed method consistently show better reconstruction results from single view depth completion to multiple views.

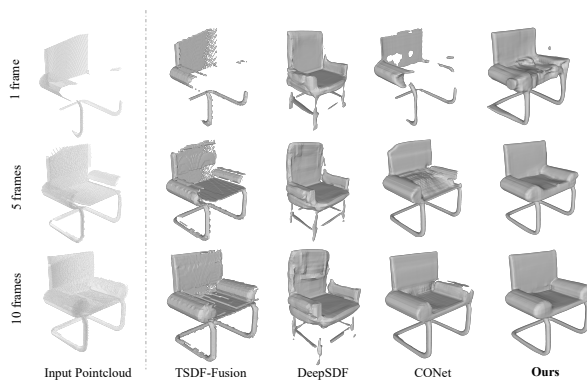


Fig. 5. Qualitative Results on reconstructions. Our method is superior to all other methods in completing missing information and reconstructing fine details.

metric loss between two frames. We re-parametrised it for object tracking.

- Point-to-Plane ICP: a non-learning geometric registration method [39]
- Color ICP: a non-learning registration method using both color and geometric information [40]
- Prior: a state-of-the-art object pose estimation using DeepSDF shape prior model. It is originally proposed in [15] for static object pose estimation and we re-parametrised it for estimating moving objects. This is equivalent to an ablation study of using the prior model only without conditioned completion refinement from Section III-C c).
- NOCS: a state-of-the-art learning-based canonical correspondence regression method. We adopted the network architecture proposed in [30]. This is equivalent to an ablation study of using only the initial prediction from

Section III-C a).

3) *Metrics*: To quantitatively evaluate the accuracy of the object tracking, we use the following metrics:

- ATE: Absolute. Trajectory Error defined in [41], in the unit of of meters
- RPE.t: relative pose error (RPE) metrics in translation defined in [41], in the unit of of metres
- RPE.R: relative pose error (RPE) metrics in rotation defined in [41], in the unit of of degrees
- R\_err: mean orientation error on each frame individually, in the unit of degrees
- t\_err: mean translation error on each frame individually, in the unit of metres

The above metrics all indicate better tracking performance when the values are lower. To analyse the trajectory, we align the first frame of the estimated object pose to the ground-truth canonical space.

4) *Results and Discussions*: Table I reports the experimental results. It shows that our approach consistently outperforms both the learning-based and non-learning-based methods in small and large motion situations. For non-learning approaches, RGB-D VO [38], Point-to-Plane ICP [39], and Color ICP [40] only leverage the depth and intensity information from two-view measurements, without taking into account any object shape prior information. The single view canonical correspondence prediction from NOCS [30] only considers shape prior information and does not take advantage of the multiview constraint. Our proposed method instead combines both multi-view constraint and shape prior information into object pose estimation for better tracking accuracy. Similar to ours, the shape prior method [15] adopts category-level shape prior from DeepSDF [16] and uses differential rendering to estimate object poses. However, latent code optimisation cannot necessarily capture the geometry deviation between training space and test shapes and thus affects the accuracy of pose estimation.

### C. Timing analysis

We implemented our system in PyTorch. The average inference time for a pair of RGB-D image in the resolution of  $320 \times 240$  is 1.337s on a RTX 3090 platform. A more-detailed breakdown of computation time for each component is shown in Table IIa. A further breakdown of computation time on tracking components is shown in Table IIb.

We would like to highlight that our current implementation is prototyped in Python. We believe a real-time system can be achieved by exploiting C++ and further GPU parallelisation.

### D. Qualitative Evaluations

We further demonstrate our proposed method in various real-world scenarios. Figure 6 shows the results in two different scenes. For each input image, we provide object reconstructions from the currently estimated camera viewpoint to visualise the observed part and from the top-down viewpoint to visualise the hidden part. As a qualitative comparison, we also show the reconstructions using classic

method [unit]	ATE [m]	RPE.t [m]	RPE.R [°]	R_err [°]	t_err [m]
<b>Ours</b>	<b>0.030</b>	<b>0.027</b>	<b>3.845</b>	<b>3.931</b>	<b>0.040</b>
Prior	0.044	0.047	8.200	6.269	0.068
RGBD	0.254	0.106	18.47	32.25	0.314
Point2Plane	0.047	0.035	4.672	5.970	0.064
Color ICP	0.254	0.114	29.12	56.18	0.320
NOCS	0.074	0.059	23.46	37.87	0.085

(a) Keyframe gap-1

method [unit]	ATE [m]	RPE.t [m]	RPE.R [°]	R_err [°]	t_err [m]
<b>Ours</b>	<b>0.033</b>	<b>0.032</b>	<b>5.243</b>	<b>4.224</b>	<b>0.043</b>
Prior	0.046	0.052	11.91	8.063	0.063
RGBD	1.068	0.403	30.89	50.95	1.309
Point2Plane	0.070	0.056	8.570	9.384	0.087
Color ICP	0.536	0.351	36.69	60.56	0.568
NOCS	0.074	0.074	21.65	37.78	0.084

(b) Keyframe gap-2

method [unit]	ATE [m]	RPE.t [m]	RPE.R [°]	R_err [°]	t_err [m]
<b>Ours</b>	<b>0.034</b>	<b>0.038</b>	<b>6.767</b>	<b>4.834</b>	<b>0.044</b>
Prior	0.043	0.050	17.20	9.885	0.061
RGBD	1.942	0.866	43.34	68.86	2.177
Point2Plane	0.807	0.442	18.22	20.16	0.892
Color ICP	2.786	1.885	42.73	77.89	2.802
NOCS	0.073	0.085	26.95	35.41	0.083

(c) Keyframe gap-4

TABLE I

QUANTITATIVE EVALUATION OF OBJECT TRACKING METHOD ON THE SYNTHETIC MOVING OBJECTS DATASET.

Components	Tracking	Integration	Completion (visualization)
Time (s)	1.284	0.003	0.474

(a) System components

Components	Initialization	Coarse est.	Dense refinement
Time (ms)	0.643	0.150	1.129

(b) Object tracking components

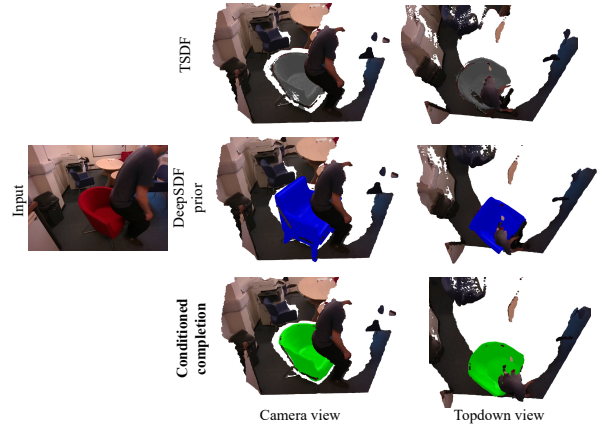
TABLE II

RUN-TIME ANALYSIS (S)

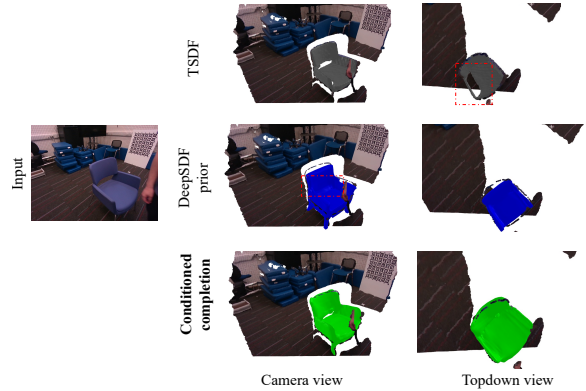
TSDF fusion [3] and the learned category-level DeepSDF object prior [16], it can be seen that TSDF-Fusion can only reconstruct the visible parts, leaving many empty holes in the object models. DeepSDF, on the other hand, has watertight reconstructions, but does not match the measurement necessarily, especially for the objects that deviate from the training space. On the contrary, our system can maintain highly detailed reconstructions and generate watertight meshes by filling in the missing parts using category-level shape priors thanks to the conditioned completion. Figure 7 also shows a scene where our system can reconstruct the visible parts and complete the hidden information of a moving object. The object pose and object geometry for the moving object are optimised jointly.

## V. CONCLUSIONS

We present a novel approach for object-level tracking and mapping system in dynamic scenes by incorporating learned category-level shape priors. It enables to reasonably



(a) Completion of a red chair



(b) Completion of a blue chair

Figure 6. Qualitative comparison of classic TSDF volume representation (gray), DeepSDF shape prior representation (blue), and our conditioned completion representation (green): our representation can reconstruct the observed part more correctly than a shape prior and completes the unseen part where TSDF fusion fails.

complete the object geometry of unseen parts based on the prior knowledge, and provide more robust and accurate tracking accuracy, even under large frame-by-frame motion and in dynamic environments with moving human involved. Experimental results in various scenarios demonstrate the effectiveness of our method. We hope our method can create a deeper understanding of exploring object prior information in object-level SLAM and benefit robots interacting with their surrounding environments. Continue from here, we would like to extend our system to a full graph-based object SLAM system.

## ACKNOWLEDGMENT

We thank Xingxing Zuo for fruitful discussions. This research is supported by Imperial College London, Technical University of Munich, and the EPSRC grant ORCA Stream B - Towards Resident Robots. Binbin Xu holds a China Scholarship Council-Imperial Scholarship.

## REFERENCES

- [1] A. J. Davison, N. D. Molton, I. Reid, and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 29, no. 6, pp. 1052–1067, 2007.

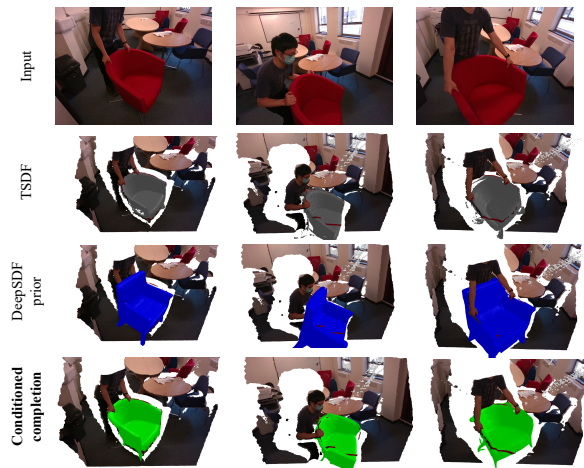


Fig. 7. Segment, track, reconstruct and complete a moving chair. Background pointclouds are just for visualization.

[2] G. Klein and D. W. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," in *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007.

[3] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-Time Dense Surface Mapping and Tracking," in *ISMAR*, 2011.

[4] E. Vespa, N. Nikolov, M. Grimm, L. Nardi, P. H. Kelly, and S. Leutenegger, "Efficient octree-based volumetric SLAM supporting signed-distance and occupancy mapping," *IEEE Robotics and Automation Letters*, 2018.

[5] J. McCormac, A. Handa, A. J. Davison, and S. Leutenegger, "SemanticFusion: Dense 3D semantic mapping with convolutional neural networks," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[6] J. McCormac, R. Clark, M. Bloesch, A. J. Davison, and S. Leutenegger, "Fusion++: volumetric object-level slam," in *Proceedings of the International Conference on 3D Vision (3DV)*, 2018.

[7] R. A. Newcombe, D. Fox, and S. M. Seitz, "Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[8] M. Jaimez, C. Kerl, J. Gonzalez-Jimenez, and D. Cremers, "Fast odometry and scene flow from rgb-d cameras based on geometric clustering," in *ICRA*, 2017.

[9] R. Scona, M. Jaimez, Y. R. Petillot, M. Fallon, and D. Cremers, "StaticFusion: Background reconstruction for dense rgb-d slam in dynamic environments," in *ICRA*, 2018.

[10] B. Bescós, J. M. Fàcil, J. Civera, and J. Neira, "DynaSLAM: Tracking, mapping and inpainting in dynamic scenes," *IEEE Robotics and Automation Letters*, 2018.

[11] M. Runz, M. Buffier, and L. Agapito, "Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects," in *ISMAR*, 2018.

[12] B. Xu, W. Li, D. Tzoumanikas, M. Bloesch, A. Davison, and S. Leutenegger, "MID-Fusion: Octree-based object-level multi-instance dynamic slam," in *ICRA*, 2019.

[13] B. Bescós, C. Campos, J. D. Tardós, and J. Neira, "DynaSLAM II: tightly-coupled multi-object tracking and SLAM," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5191–5198, 2021.

[14] E. Sucar, K. Wada, and A. J. Davison, "NodeSLAM: Neural object descriptors for multi-view shape reconstruction," in *Proceedings of the International Conference on 3D Vision (3DV)*, 2020.

[15] J. Wang, M. Rünz, and L. Agapito, "Dsp-slam: Object oriented slam with deep shape priors," in *Proceedings of the International Conference on 3D Vision (3DV)*, 2021.

[16] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "DeepSDF: Learning continuous signed distance functions for shape representation," in *CVPR*, 2019.

[17] C.-C. Wang, C. Thorpe, and S. Thrun, "Online simultaneous local-

ization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded urban areas," in *ICRA*, 2003.

[18] M. Rünz and L. Agapito, "Co-fusion: Real-time segmentation, tracking and fusion of multiple objects," in *ICRA*, 2017.

[19] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "ElasticFusion: Real-time dense SLAM and light source estimation," *International Journal of Robotics Research (IJRR)*, vol. 35, no. 14, pp. 1697–1716, 2016.

[20] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics (T-RO)*, vol. 33, no. 5, pp. 1255–1262, 2017.

[21] M. Strecke and J. Stückler, "Em-fusion: Dynamic object-level slam with probabilistic data association," in *CVPR*, 2019.

[22] —, "Where does it end?-reasoning about hidden surfaces by object intersection constraints," in *CVPR*, 2020.

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

[24] R. Wang, N. Yang, J. Stückler, and D. Cremers, "Directshape: Photometric alignment of shape priors for visual vehicle pose and shape estimation," in *ICRA*, 2020.

[25] L. Hu, W. Xu, K. Huang, and L. Kneip, "Deep-slam++: Object-level rgbd slam based on class-specific deep shape priors," *arXiv preprint arXiv:1907.09691*, 2019.

[26] K. Li, H. Rezaatofghi, and I. Reid, "Moltr: Multiple object localization, tracking and reconstruction from monocular rgb videos," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3341–3348, 2021.

[27] B. Xu, A. J. Davison, and S. Leutenegger, "Deep probabilistic feature-metric tracking," *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 223 – 230, 2021.

[28] B. Wen and K. E. Bekris, "Bundletrack: 6d pose tracking for novel objects without instance or category-level 3d models," in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2021.

[29] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6d object pose and size estimation," in *CVPR*, 2019.

[30] D. Rempe, T. Birdal, Y. Zhao, Z. Gojic, S. Sridhar, and L. J. Guibas, "Caspr: Learning canonical spatiotemporal point cloud representations," *Neural Information Processing Systems (NIPS)*, 2020.

[31] N. Muller, Y.-S. Wong, N. J. Mitra, A. Dai, and M. Nießner, "Seeing behind objects for 3d multi-object tracking in rgb-d sequences," in *CVPR*, 2021.

[32] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2016, pp. 424–432.

[33] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[34] A. Majercik, C. Crassin, P. Shirley, and M. McGuire, "A ray-box intersection algorithm and efficient dynamic voxel rendering," *Journal of Computer Graphics Techniques Vol.*, vol. 7, no. 3, pp. 66–81, 2018.

[35] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 13, no. 04, pp. 376–380, 1991.

[36] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[37] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, "Implicit geometric regularization for learning shapes," *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

[38] F. Steinbrücker, J. Sturm, and D. Cremers, "Real-Time Visual Odometry from Dense RGB-D Images," in *Workshop on Live Dense Reconstruction from Moving Cameras at ICCV*, 2011.

[39] S. Rusinkiewicz and M. Levoy, "Efficient Variants of the ICP Algorithm," in *Proceedings of the IEEE International Workshop on 3D Digital Imaging and Modeling (3DIM)*, 2001.

[40] J. Park, Q.-Y. Zhou, and V. Koltun, "Colored point cloud registration revisited," in *ICCV*, 2017.

[41] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A Benchmark for the Evaluation of RGB-D SLAM Systems," in *IROS*, 2012.



# Appendix: Learning to Complete Object Shapes for Object-level Mapping in Dynamic Scenes

Binbin Xu<sup>1</sup>, Andrew J. Davison<sup>1</sup>, Stefan Leutenegger<sup>1,2</sup>

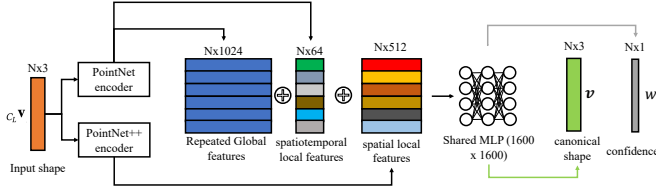


Fig. 1. The architecture of our canonical correspondence network. The architecture is modified from [1]. It extracts global features and spatiotemporal local features from the PointNet encoder [2] and spatial local features from the PointNet++ encoder [3]. These features are concatenated and passed to an MLP to regress the canonical shape correspondence and the associated confidence.

The learnable network parameters in this work include three parts, canonical correspondence network  $F_n$ , shape prior network  $F_0$ , and shape posterior network  $F_1$ .

Figure 1 shows the architecture of our canonical correspondence network  $F_n$ . It takes the partial pointcloud  $C_L \mathbf{v}$  from the depth measurements as input and predicts its correspondence  $\mathbf{v}$  in canonical space and the associated confidence  $w$ . We train the canonical correspondence network using partial pointclouds generated from the synthetic shapenet dataset [4]. During the training, we augment the input pointcloud with random object poses and solve the 7DoF object poses using Equation (1). To help network prediction robust to outliers, we also add random depth outliers in the pointcloud generation to learn the correspondence confidence  $w$  in a self-supervised way. The solved pose is compared to the augmented ground-truth pose and the whole network is trained end-to-end since the estimation is differentiable.

$$\operatorname{argmin}_{s_{O_n}, \mathbf{T}_{C_L O_n}, \mathbf{u}_L \in M_n} w \left( \mathbf{v} - \frac{1}{s_{O_n}} \mathbf{T}_{C_L O_n}^{-1} C_L \mathbf{v}(\mathbf{u}_L) \right). \quad (1)$$

We use the pre-trained off-shelf network weights from the category-level shape prior network DeepSDF [5] for  $F_0$ , which was also trained in the synthetic shapenet dataset [4]. Its architecture is visualized in Figure 2.

Figure 3 shows the architecture of our posterior shape completion network  $F_1$ . It takes the input of a TSDF feature volume and a TSDF confidence volume, which are extracted separately from a partial TSDF volume and its weight volume. The input of TSDF confidence volume is designed to balance the observed depth measurement and shape prior

The authors are with Department of Computing, Imperial College London, United Kingdom. {b.xu17, a.davison, s.leutenegger}@imperial.ac.uk

<sup>2</sup> The author is also with the Smart Robotics Lab, Technical University of Munich, Germany {stefan.leutenegger}@tum.de

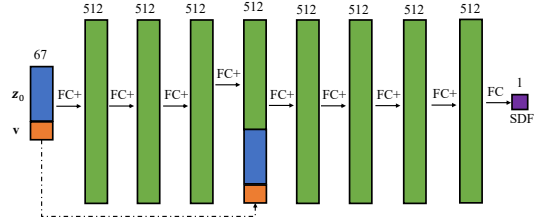


Fig. 2. The architecture of the shape prior network [5]. The input vector is fed through a decoder, which contains eight fully-connected (FC) layers with one skip connection. FC+ denotes a FC with a following softplus activation and the last FC layer output a single SDF value.

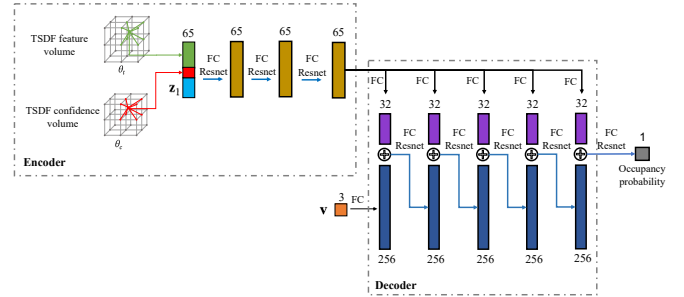


Fig. 3. The architecture of our shape completion network, modified from CONet [6]. The encoder extracts the TSDF feature vector  $\theta_f[\mathbf{v}] \in \mathbb{R}^{32}$  and the TSDF confidence vector  $\theta_c[\mathbf{v}] \in \mathbb{R}^1$  from TSDF feature volume and TSDF confidence volume, respectively, and concatenates them with a latent code  $\mathbf{z}_1$  as an input to the network. It goes through 3 fully-connected (FC) ResNet-blocks to extract local latent features, which are then fed into an occupancy decoder [7] to predict occupancy probabilities on the position vector  $\mathbf{v}$ .

information. The unseen part has TSDF weight of zero value and biases towards prior shape and will gradually switch to 3D reconstruction when more depth information is fused into the corresponding TSDF voxel. We additionally concatenate the inputs with a latent code  $\mathbf{z}_1 \in \mathbb{R}^{32}$  so that the hidden space can be optimised to generate novel shapes, as shown in Figure 4. The shape completion network  $F_1$  can predict a complete object geometry represented in a continuous occupancy function by inferring an occupancy value  $o$  on any given 3D position  $\mathbf{v} \in \mathbb{R}^3$  in canonical space.

Since the partial observation in reality mostly happens due to self-occlusions and sometimes also due to occlusions from other objects, we rendered depth maps using objects in the shapenet dataset [4]. To train the posterior shape completion network, we rendered depth maps for each object in the shapenet dataset [4] to simulate partial depth observations. We use the occupancy loss defined in Equation (2) to encourage the completed shape to be similar to the ground-

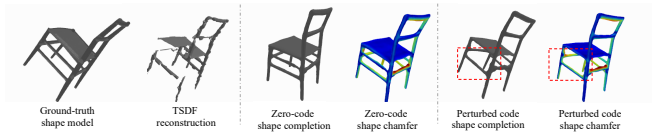


Fig. 4. Editing the conditioned latent code can change the geometry of the unobserved part in the object model.

truth one.

$$E_{\text{occ}} = -\sum_{\mathbf{v}} [o_{\mathbf{v}} \log(o_{\mathbf{v}}^*) + (1 - o_{\mathbf{v}}) \log(1 - o_{\mathbf{v}}^*)]. \quad (2)$$

Similar to the training in DeepSDF [5], different object shapes belonging to the same category have different latent codes, but share the same decoder network weight. We make different partial observations of the same object shape share the same latent code.

#### REFERENCES

- [1] D. Rempe, T. Birdal, Y. Zhao, Z. Gojcic, S. Sridhar, and L. J. Guibas, “Caspr: Learning canonical spatiotemporal point cloud representations,” *Neural Information Processing Systems (NIPS)*, 2020.
- [2] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation,” in *CVPR*, 2017, pp. 652–660.
- [3] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space,” in *Neural Information Processing Systems (NIPS)*, 2017, pp. 5099–5108.
- [4] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, “ShapeNet: An information-rich 3d model repository,” *arXiv preprint arXiv:1512.03012*, 2015.
- [5] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, “DeepSDF: Learning continuous signed distance functions for shape representation,” in *CVPR*, 2019.
- [6] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, “Convolutional occupancy networks,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [7] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, “Occupancy networks: Learning 3d reconstruction in function space,” in *CVPR*, 2019.