CIRCLE: Convolutional Implicit Reconstruction and Completion for Large-scale Indoor Scene

Hao-Xiang Chen, Jiahui Huang, Tai-Jiang Mu*, and Shi-Min Hu

BNRist, Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
{chx20,huang-jh18}@mails.tsinghua.edu.cn
{taijiang,shimin}@tsinghua.edu.cn

Abstract. We present CIRCLE, a framework for large-scale scene completion and geometric refinement based on local implicit signed distance functions. It is based on an end-to-end sparse convolutional network, CircNet, which jointly models local geometric details and global scene structural contexts, allowing it to preserve fine-grained object detail while recovering missing regions commonly arising in traditional 3D scene data. A novel differentiable rendering module further enables a test-time refinement for better reconstruction quality. Extensive experiments on both real-world and synthetic datasets show that our concise framework is effective, achieving better reconstruction quality while being significantly faster.

Keywords: Scene Reconstruction, Scene Completion, Differentiable rendering

1 Introduction

In recent years, 3D reconstruction from RGB-D camera data has been widely explored thanks to its ease of acquisition with many applications in robotic perception, virtual reality, games, *etc.* It is well-accepted that an ideal reconstruction algorithm should be capable of simultaneously (i) restoring fine-grained geometric details in the target scene, (ii) handling with large scenes efficiently, and (iii) completing the missing regions. Additionally, the underlying 3D representation should be flexible enough to allow further optimization of geometric quality.

However, traditional algorithms along with their accompanying representations fail to effectively fulfil the above requirements. For instance, methods using the *truncated signed distance function* (TSDF) [10,34] are hampered by limited voxel resolution and lack robustness to noisy data. Surfels [52] offer more flexibility by treating the 3D scene as unstructured points, but maintaining correct topology is challenging. Furthermore, these methods cannot fill in missing geometry in the scene, which is common in practice due to the sensor limitations, incomplete coverage of the scanning trajectory, or unreachable areas.

* Corresponding author.

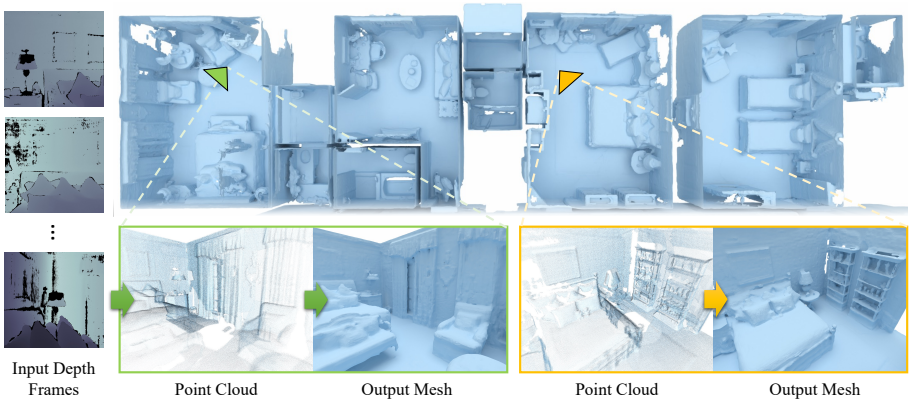

Fig. 1: **CIRCLE** efficiently reconstructs and completes 3D scene from a given a sequence of posed depth images, commonly corrupted by noise and missing data: the inference time for this scene takes only 17s, 10× faster than [39].

The recent introduction of deep implicit representations [7,32,38] has enabled a plethora of research directions for 2D and 3D data processing. Parameterized by a neural network, implicit functions are inherently continuous and differentiable. Notably, in the field of 3D reconstruction, various works [8,9,16,17,36,44] have already demonstrate their ability to learn *object-level* geometric priors from the shape repositories. However, when applied to large-scale scenes, the above methods are typically impractical. The reasons are three-fold. Firstly, the structure of a scene is substantially more complicated than that of a single object. A typical end-to-end, optimization-free, framework is weak at capturing the entangled geometric priors of cluttered regions. Secondly, though there exist other work [1,5,23,43,48] that overfits the scene geometry, to avoid the necessity of prior learning, it usually involves costly optimization procedures. Thirdly, some efforts [22,39,46,47] have been made to reconstruct scenes in real-time with deep implicit functions, but performed at a cost of low reconstruction quality.

To tackle these issues, we introduce the CIRCLE framework, as shown in Fig. 1. It employs a novel CircNet, short for fully-convolutional implicit network for reconstruction and completion of large-scale indoor 3D scenes from partial point clouds. It is capable of both preserving scene geometric details and completing missing regions of the scene in a semantically-meaningful way. Specifically, we adopt a local implicit grid to represent the local details of the whole scene, and learn the global contextual information for scene completion via a sparse U-Net. Our network is also efficient, in that it encodes and decodes the sparsity pattern of the scene geometry by learning, and only non-empty portions need to be evaluated. Furthermore, we provide a fast and novel differentiable rendering approach tailored for refining our output representation, which can greatly improve the geometric quality during inference to provide resilience to the errors in the raw input. Extensive experiments on various datasets demonstrate

the effectiveness of our framework, which sets a new state-of-the-art for scene reconstruction and completion.

2 Related Work

Scene Reconstruction. Building a high-quality and coherent scene-level reconstruction is challenging due to noise, occlusion and missing data inherent in 3D data acquisition sensors. While traditional methods [4,13,30,34,37,41,52,53] incrementally fuse input depth observations using a moving average [10], learning methods [50,51] can further reduce the noise using data-driven geometric biases. The recent trend of using implicit neural representations, such as DI-Fusion [22] and its successors [3,46], either uses localized priors or the continuous nature of a globally-supported network function. In comparison, our method can not only accurately recover detailed scene geometry, but also rebuild missing parts via a global structural reasoning based on learning. We refer readers to [26] for a more comprehensive understanding on scene reconstruction.

Scene Completion. The main challenge in scene completion is to fill missing regions with data that are semantically coherent with the existing content. [45] casts the problem in terms of panoramic image completion but important geometric details are significantly missing. [14] first brings the aid of semantic segmentation to the completion problem in the 3D domain. Subsequent lines of work [12,15] tackle the problems of geometric sparsity and color generation. We note that many end-to-end frameworks [1,39] using implicit representations also provide decent scene extrapolation due to the continuous nature of networks, even though they are not specifically designed for this task.

Differentiable Rendering. The technique of differentiating the rendering process bridges the gap between 3D geometry and 2D observations of it by allowing for end-to-end optimization directly from the captured raw sensor data, which was first applied to triangular meshes [25,29] and later to implicit fields [24,28,35]. The prevalence of NeRF [33] motivates many studies to improve rendering efficiency and fitting speed, either through localized structures [27], level-of-detail rendering [48], caching [55], or multi-view stereo [42,54]. In conjunction with our novel local implicit representation, we devise a new differentiable rendering approach which can rapidly and effectively refine the geometric details of the reconstructed scene during inference.

3 CIRCLE: Convolutional Implicit Scene Reconstruction and Completion

Problem Formulation. The input to our method is a sequence of posed depth frames $\{\mathcal{D}_t, \mathbf{T}_t\}_{t=1}^{T}$, with $\mathcal{D}_t \in \mathbb{R}^{W \times H}$ and $\mathbf{T}_t \in \mathbb{SE}(3)$ being the depth image and the 6-DoF camera pose, respectively. Our goal is to build a high-quality and complete 3D reconstruction of the scene, represented using M a local sparse

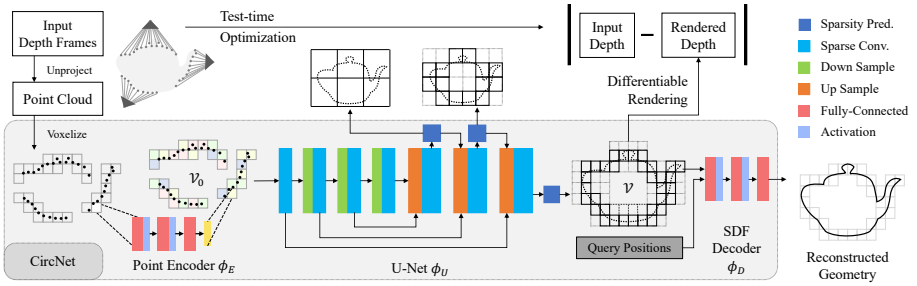

Fig. 2: **Pipeline.** We first voxelize the accumulated unprojected points from the input posed depth frames into a sparse grid. The feature volume is then fed to CircNet, which comprises 3 neural networks: ϕ_{E}, ϕ_{U}, and ϕ_{D}, and outputs an implicit completed surface. Inference-time refinement is enabled by our differentiable rendering, aiming at better and more complete reconstruction.

implicit voxel grid $\mathcal{V} = \{(\boldsymbol{c}_m, \boldsymbol{l}_m)\}_{m=1}^{M}$ that contain the surface of the scene geometry. Here, $\boldsymbol{c}_m \in \mathbb{R}^3$ is the voxel coordinate and $\boldsymbol{l}_m \in \mathbb{R}^L$ is the latent vector describing the local voxel grid's geometry, from which we can decode the signed distance values of the full scene and finally extract the mesh. The size of each voxel is $b \times b \times b$.

Overview. As Fig. 2 shows, we first unproject all the depths $\mathcal{D}_t$ under the given poses $\mathbf{T}_t$ to obtain an accumulated point cloud $\mathcal{P} = \{(\boldsymbol{p}_i, \boldsymbol{n}_i)\}_{i=1}^{N}$, where $\boldsymbol{p}_i \in \mathbb{R}^3$ and $\boldsymbol{n}_i \in \mathbb{R}^3$ are the point position and its estimated normal, using [34]. $\mathcal{P}$ is then voxelized into initial sparse 3D grid and processed by CircNet (see Sec. 3.1). Being aware of both global scene structure and local geometric details, CircNet simultaneously refines the voxelized points and adds additional points using a point encoder ϕ_{E} and a U-Net ϕ_{U}, and produces $\mathcal{V}$ defining the latent vector of local implicit geometry, which is then decoded to TSDF values by a multi-layer perceptron (MLP) ϕ_{D}. One can later extract the mesh using marching cubes [31] from these TSDF values. Moreover, the reconstructed geometry can be further optimized during inference-time via a novel differentiable rendering scheme described in Sec. 3.2, to refine both the scene geometry and the camera pose. Detailed loss functions for the training procedure and the inference-time refinement are discussed in Sec. 3.3.

3.1 CircNet Architecture

Given the unprojected point cloud $\mathcal{P}$ from the input views, CircNet sequentially applies three trainable components: a point encoder network ϕ_{E}, a U-Net ϕ_{U}, and an SDF decoder ϕ_{D} to produce an implicit representation of the underlying scene. We now describe these components in turn.

Point Encoder. We first split the input point cloud into voxels. For a point $\boldsymbol{p}_i$, the index of its belonging voxel m_i is determined by the m satisfying $\boldsymbol{p}_i \in$

$[\boldsymbol{c}_m, \boldsymbol{c}_m + b)$. We define the local coordinate of $\boldsymbol{p}_i$ within its belonging voxel as $\boldsymbol{p}_i^l = (\boldsymbol{p}_i - \boldsymbol{c}_m)/b \in [0,1]^3$. Next, for each voxel m, we feed all the local coordinates of points contained in the voxel, along with their normals: $\{(\boldsymbol{p}_i^l, \boldsymbol{n}_i) \in \mathbb{R}^6 \mid m_i = m\}$ into a point encoder ϕ_{E}. ϕ_{E} adopts a basic PointNet [40] structure by first mapping all the input features into L-dimensions with a shared MLP and then aggregating the features via mean pooling. The resulting sparse feature voxel grid is denoted by $\mathcal{V}_0$.

U-Net. The goal of the U-Net ϕ_{U} in this step is to complete and refine the reconstruction from $\mathcal{V}_0$ into $\mathcal{V}$. This is achieved by propagating contextual features in the hierarchical U-Net structure with a large receptive field. A trivial implementation falls back to a dense convolution that generates a dense feature grid even if many voxels are actually empty. Due to the sparse nature of the geometry, we instead use submanifold sparse convolution [18] for our convolution layer. For the decoder branch, inspired by [49], we append a *sparsity prediction* module to each layer of the decoder. This module is instantiated with a shared MLP applied to each voxel and predicts the confidence of the current voxel containing the true surfaces; voxels with scores lower than 0.5 are pruned. Accordingly, usual skip connections are replaced by sparsity-guided skip connections: connections are only added for voxels predicted to be non-empty. Apart from the efficiency gain, this design also eases network training by obviating the need to model the full geometry of empty regions.

SDF Decoder. To recover the final scene geometry, we traverse all points $\boldsymbol{p}$ in the non-empty regions of $\mathcal{V}$ and learn the signed distance values using an implicit decoder instantiated with an MLP $\phi_{\mathrm{D}} : (\boldsymbol{p}^l, \hat{\boldsymbol{l}}) \in \mathbb{R}^{3+L} \mapsto [-1, 1]$, where $\boldsymbol{p}^l$ is the local coordinate of $\boldsymbol{p}$ and $\hat{\boldsymbol{l}}$ is the interpolated feature taken from $\mathcal{V}$. To smooth the geometric interpolation across voxel boundaries, we apply an additional $2 \times 2 \times 2$ convolution over $\mathcal{V}$ to propagate the features stored at voxel *centers* to voxel *corners*, obtaining $\{\boldsymbol{l}'_m\}$. The input feature $\hat{\boldsymbol{l}}$ can then be trilinearly interpolated $\psi(\cdot)$ from the features stored at its 8 nearest voxel corners: $\hat{\boldsymbol{l}} = \psi(\boldsymbol{p}^l, \{\boldsymbol{l}'_{(1)}, \ldots, \boldsymbol{l}'_{(8)}\})$.

3.2 Differentiable Local Implicit Rendering

Despite the good-quality, end-to-end reconstruction provided by CircNet, some desired geometric details can be lost. The reasons are two-fold. Firstly, the real-world depth acquisition usually suffers from noisy pose and sensor limitations, resulting in erroneous reconstruction and severe missing regions. Secondly, a simple feed-forward network trained on large-scale datasets can underfit geometric features or generate excessive contents [32,38]. Being aware of these issues, we propose a novel differentiable renderer for our implicit representation, allowing for effective differentiation through both geometry and camera pose. Specifically, for each pixel to be rendered, we emit a ray with an origin $\boldsymbol{o}$ and a unit direction $\boldsymbol{d}$, and compute the depth of the intersection z so that the intersection point is $\boldsymbol{p} = \boldsymbol{o} + z\boldsymbol{d}$. The forward and backward passes are defined as follows:

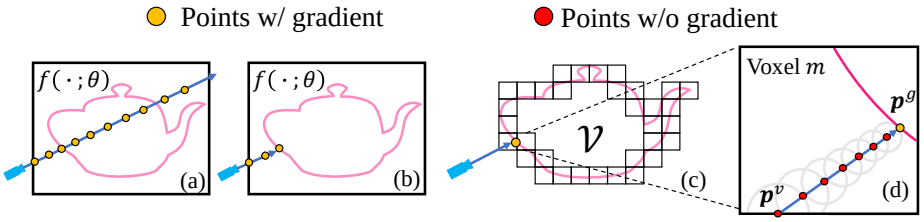

Fig. 3: **Rendering strategies.** Rendering a globally-supported implicit representation with (a) uniform query points [33] and (b) differentiable sphere tracing [28]. (c,d) Our approach with both sphere tracing and implicit differentiation, only computes gradients for on-surface points, thus being highly efficient.

Forward Pass. The forward pass is composed of two steps as shown in Fig. 3 (c–d):

1. **Voxel-level Intersection.** As the sparsity prediction modules from the different layers of our U-Net decoder naturally form an *octree* structure thanks to the upsampling operator, we can use any existing ray-octree intersection algorithm for this step. In our implementation, we choose the fast algorithm in [48] that generates a list of intersection pairs $\{(z^v, m^v)\}$, where z^v is the depth and m^v is the voxel index of the intersection.
2. **Geometry-level Intersection.** The sphere tracing algorithm [20] is applied for each intersecting voxel m^v, starting from $\boldsymbol{o} + z^v\boldsymbol{d}$ and ending at $\boldsymbol{p}^g = \boldsymbol{o} + z^g\boldsymbol{d}$ that hits the surface. Note that only the smallest z^g among all the voxels is returned as the final depth z due to occlusion.

Backward Pass. For clarity, we abstract our full CircNet as an implicit network $f(\boldsymbol{p};\theta)$ whose inputs are the position $\boldsymbol{p}$ and the intermediate features or network parameters θ, and the output is the signed distance value. We wish to compute the first-order derivative of the depth z w.r.t.θ as well as the camera ray $\boldsymbol{o}$ and $\boldsymbol{d}$ for optimization. Inspired by [54], we employ the fact that $f(\boldsymbol{o} + z\boldsymbol{d};\theta) \equiv 0$ and use implicit differentiation to obtain:

$$\frac{\partial z}{\partial \theta} = -\gamma\frac{\partial f}{\partial \theta}, \quad \frac{\partial z}{\partial \boldsymbol{o}} = -\gamma\frac{\partial f}{\partial \boldsymbol{p}}, \quad \frac{\partial z}{\partial \boldsymbol{d}} = -\gamma z\frac{\partial f}{\partial \boldsymbol{p}}, \tag{1}$$

where $\gamma = \langle \boldsymbol{d}, \partial f/\partial \boldsymbol{p}\rangle^{-1}$ is a scalar, $\langle\cdot,\cdot\rangle$ denotes the vector inner product, and other derivatives related to f can be efficiently evaluated using reverse-mode back-propagation. Empirically, we observe that full gradient-based optimization over all network parameters fails to converge. Hence we choose to only optimize the latent vectors in $\mathcal{V}$: $\theta = \{\boldsymbol{l}_m\}$, and fix all other parts of the networks.

Discussion. A comparison between our method and previous approaches is shown in Fig. 3. Methods similar to, *e.g.*, NeRF [33] exhaustively query all points along the ray; most of the unnecessary computations far away from the surface can be saved with sphere tracing [20,28]. Our use of *localized* grid further speed up the process thanks to the explicit ray-voxel intersection step that greatly

reduces the number of steps in tracing. Nevertheless, a naive implementation of the backward pass requires unrolling the tracing steps, leading to inaccurate gradients. We for the first time marry the merits of implicit differentiation, originally designed for global representations [54], with our local feature grid, so that only the intersection points need to be stored in the computation graph, leading to a fast, stable, accurate and memory-efficient method for both forward and backward passes. Experiments verifying our design choices are shown in Sec. 4.3.

3.3 Loss Functions

CircNet Loss Function. The three networks ϕ_{E}, ϕ_{U} and ϕ_{D} are jointly trained in an end-to-end manner, using the following loss function:

$$\mathcal{L} = \mathcal{L}_{\mathrm{sdf}} + \alpha\mathcal{L}_{\mathrm{norm}} + \beta\mathcal{L}_{\mathrm{struct}} + \delta \sum_{m=1}^{M} \|\boldsymbol{l}_m\|, \tag{2}$$

where $\|\cdot\|$ is the vector norm. $\mathcal{L}_{\mathrm{sdf}}$ is the data term defined as the L1 distance between the predicted signed distance from the decoder $\phi_{\mathrm{D}}(\boldsymbol{p}^l, \hat{\boldsymbol{l}})$ and the ground-truth values $s^{\mathrm{gt}}(\boldsymbol{p})$:

$$\mathcal{L}_{\mathrm{sdf}} = \int_{\Omega_u \cup \Omega_n} |\phi_{\mathrm{D}}(\boldsymbol{p}^l, \hat{\boldsymbol{l}}) - s^{\mathrm{gt}}(\boldsymbol{p})| \, \mathrm{d}\boldsymbol{p}. \tag{3}$$

Here Ω_u denotes the occupied region of the voxels $\mathcal{V}$ while Ω_n is a narrow band region near the surface. The normal of the predicted geometry, computed as $\nabla_{\boldsymbol{p}}\phi_{\mathrm{D}}$, is constrained by the normal loss:

$$\mathcal{L}_{\mathrm{norm}} = \int_{\Omega_u \cup \Omega_n} \big|\|\nabla_{\boldsymbol{p}}\phi_{\mathrm{D}}\| - 1\big| \, \mathrm{d}\boldsymbol{p} + \int_{\Omega_n} \big(1 - \langle \nabla_{\boldsymbol{p}}\phi_{\mathrm{D}}, \boldsymbol{n}^{\mathrm{gt}}(\boldsymbol{p})\rangle\big) \, \mathrm{d}\boldsymbol{p}, \tag{4}$$

where the first term enforces the eikonal equation of the signed distance field while the second term minimizes the angle between predicted normal and ground-truth normal $\boldsymbol{n}^{\mathrm{gt}}$.

$\mathcal{L}_{\mathrm{struct}}$ uses cross-entropy loss to supervise the sparsity prediction module for each layer in the decoder branch of ϕ_{U}. Specifically, we obtain the ground-truth sparsity pattern of the target geometry at multiple resolutions in accordance with the output sparsity map from the U-Net, and directly supervise the predicted confidence score. During training, we use the ground-truth sparsity map instead of the predicted one for the skip-connections and pruning of the next layer.

Inference-time Refinement. During inference, our differentiable rendering module is applied to refine the predicted geometry and the camera poses. For each depth image $\mathcal{D}_t$ and its pose $\mathbf{T}_t$, we can render a depth image as $\mathcal{D}'_t(\mathbf{T}_t, \theta) \in \mathbb{R}^{W \times H}$, whose values are the depths $\{z\}$ from Sec. 3.2. By minimizing the error between the rendered depth and the observed depth, we can jointly optimize the quality of geometry and input poses:

$$\min_{\theta, \{\delta\mathbf{T}_t\}} \sum_{t=1}^{T} \big|\mathcal{D}_t - \mathcal{D}'_t(\delta\mathbf{T}_t\mathbf{T}_t, \theta))\big|, \tag{5}$$

where we optimize an increment to pose $\delta\mathbf{T}_t$ instead of $\mathbf{T}_t$ itself, for better convergence.

4 Experiments

4.1 Datasets and Settings

Datasets. The main dataset used to evaluate our framework is N-Matterport3D. Adapted from [6], this dataset contains 1,788 and 394 scans of rooms from 90 buildings for training/validation and testing, respectively, captured by a Matterport Pro Camera. We follow the self-supervised setting from [12] by randomly sampling 50% of the frames to generate an incomplete version of each room and supervise our method with a complete version reconstructed from all frames. To test the robustness of our model, we follow [50] and add synthetic noise to each individual depth frame (denoted by the prefix 'N-'), because raw MatterPort3D dataset have little noise. We additionally used the well-known ICL-NUIM [19] public benchmark containing 4 scan trajectories for testing only, to demonstrate the generalizability of our method. We also show qualitative results on ScanNetv2 [11] dataset to demonstrate how our network works under real-world noise. Due to the incomplete ground-truth meshes and inaccurate poses in ScanNetv2, it is not ideal for training. We hence directly apply our model trained on N-Matterport3D to this dataset.

Parameter Settings. Our CircNet was trained and tested on a single Nvidia GeForce RTX 2080Ti GPU. The weights of the loss terms are empirically set to $\alpha = 0.1$, $\beta = 1$ and $\delta = 0.001$. We used the Adam optimizer with a learning rate of 0.001. For efficient training, we uniformly split the input point cloud $\mathcal{P}$ into patches of size $3.2\text{m} \times 3.2\text{m} \times 3.2\text{m}$, although as a fully convolutional architecture, our pipeline could easily scale to the full scene during inference. ϕ_{E}, ϕ_{U} and ϕ_{D} have 4, 5, and 3 layers respectively. With the scale of indoor scenes, the voxel size b is set to 0.05m and the width of Ω_n is set to 2.5mm. Further details of our network structure are given in the supplementary material.

Baseline. Our method is compared to a full spectrum of methods, including those providing reconstruction from sequential depth frames, *i.e.*, RoutedFusion [50] (denoted as "R-Fusion") and DI-Fusion [22] using representations of either local implicit grid or a neural signed-distance volume. We further consider methods operating on fully-fused geometry, *i.e.*, the convolutional occupancy network [39] (denoted as "ConvON") is the state-of-art local implicit network for surface reconstruction considering global information, while SPSG [15] is the up-to-date scene completion approach that takes TSDF volumes as input. For methods that are cannot be trained on large-scale scenes, we used pre-trained weights obtained from synthetic datasets.

Metrics. We use root mean square error (RMSE), chamfer distance (CD), surface precision, recall, and F-score during evaluation. RMSE, CD, and surface precision mainly measure the accuracy of the reconstruction, surface recall mainly

Table 1: **Quantitative results using the N-Matterport3D dataset.**

	RMSE ↓ ($\times 10^{-3}$)	CD ↓ ($\times 10^{-3}$)	F-Score ↑ (%)	Precision ↑ (%)	Recall ↑ (%)
SPSG [15]	27.1	1.05	80.12	76.03	85.21
ConvON [39]	31.4	1.85	62.34	52.32	78.45
R-Fusion [50]	24.6	0.98	65.64	64.10	67.54
DI-Fusion [22]	20.9	1.14	82.36	82.31	82.68
Ours (w/o optim.)	<u>16.5</u>	<u>0.47</u>	<u>89.11</u>	<u>88.93</u>	<u>89.11</u>
Ours	**16.2**	**0.47**	**89.23**	**89.23**	**89.24**

↓ / ↑: Lower / higher is better. **Bold** numbers indicate the best and <u>underlined</u> numbers indicate the second best.

assesses the degree of completeness, and F-score reflects both accuracy and completeness. All reconstruction results from different methods are converted to point clouds for fair comparisons. RMSE and CD are measured in meters, and for precision and recall, a predicted or ground truth point is accepted if its distance to the closest ground truth or predicted point is smaller than 0.02 m.

4.2 Comparisons to Other Methods

Tab. 1 shows that our proposed method performs the best in terms of all metrics, for the N-Matterport3D dataset. Qualitative results are presented in Fig. 4. The dense structure of ConvON makes it difficult to simultaneously capture the local and global information from real-world datasets. R-Fusion and DI-Fusion only learn local geometric priors from the synthetic datasets. Specifically, although DI-Fusion fits local details with local implicit functions and achieves competitive performance, its lack of global information prevents it from completing missing regions. SPSG shows a capability for scene completion; however, limited by the discrete TSDF representation, the precision of the reconstructed surface is unsatisfactory. Our method learns global contextual information from local implicit grid by the convolutional neural network ϕ_{U}, and thus can faithfully reconstruct local geometric details as well as recovering many missing regions. As highlighted with the red boxes in Fig. 4, our network effectively fills in the holes in the planar regions (*e.g.*, walls, floors, and ceilings). Some of the objects such as beds, tables, pillows, *etc.*, can also be completed in a semantically meaningful way.

We further evaluate the generalizability of all approaches using the ICL-NUIM dataset; quantitative results are given in Tab. 2. Remarkably, although our method is trained using panoramic scans as in [6], thanks to our effective learning scheme in 3D space, it generalizes well to hand-held trajectories whose geometric distributions are drastically different.

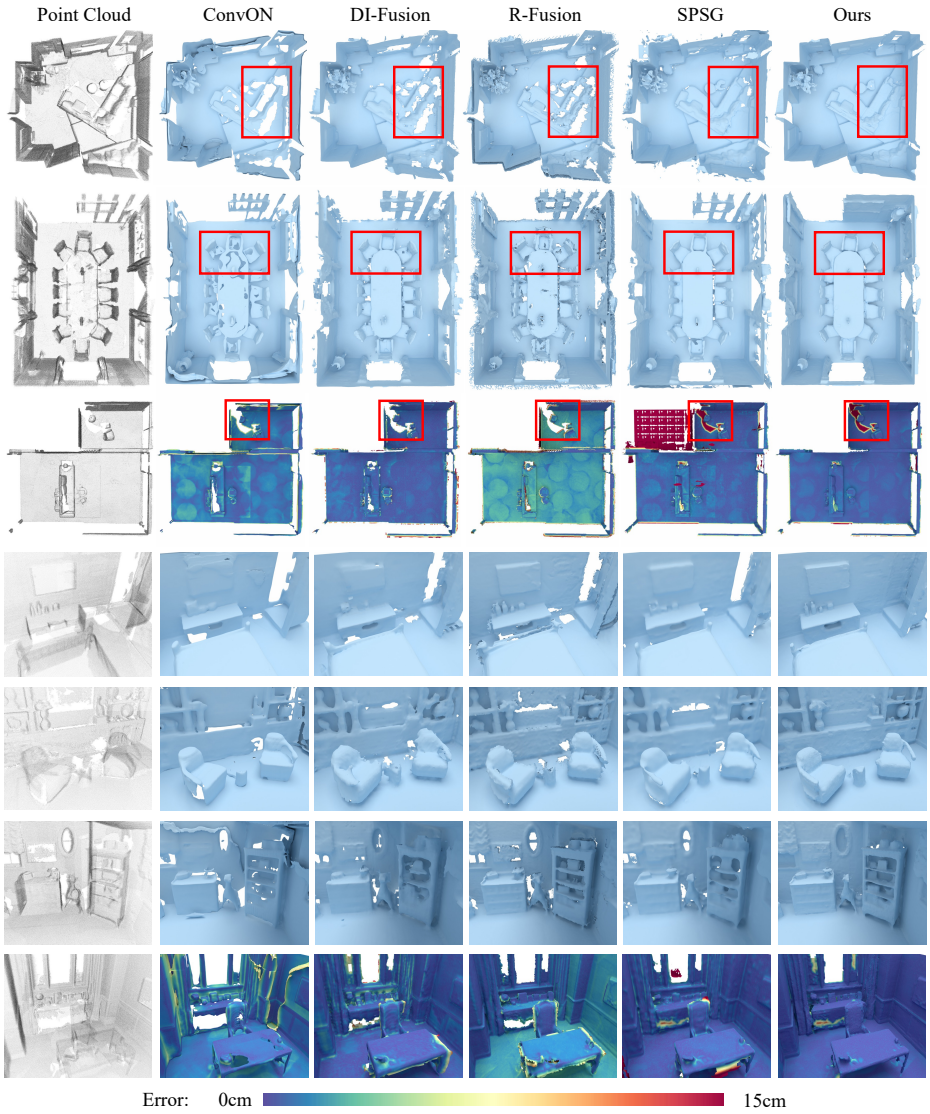

Fig. 4: **Visual comparison using N-Matterport3D.** Results show both global views (part 1, top three rows) and close-up views (part 2, bottom four rows). The last row in each part shows each method's per-point error, the distance between each reconstructed vertex and the corresponding closest ground truth point.

Table 2: **Quantitative results tested on the ICL-NUIM dataset.**

	RMSE ↓ ($\times 10^{-3}$)	CD ↓ ($\times 10^{-3}$)	F-Score ↑ (%)	Precision ↑ (%)	Recall ↑ (%)
SPSG [15]	36.6	2.07	20.29	29.70	15.62
ConvON [39]	42.3	3.55	13.81	18.46	11.15
R-Fusion [50]	40.9	2.75	14.56	22.20	11.07
DI-Fusion [22]	**19.5**	**1.32**	22.14	<u>51.21</u>	14.23
Ours (w/o optim.)	22.7	1.54	<u>23.89</u>	51.02	<u>15.78</u>
Ours	<u>22.1</u>	<u>1.46</u>	**25.54**	**53.55**	**16.99**

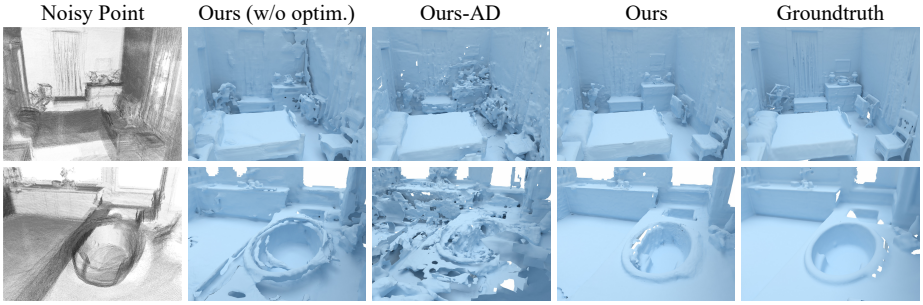

Fig. 5: **Inference-time refinement with differentiable rendering.** Our proposed method (Ours) effectively fixes the initial pose error (Ours w/o optim.) and produces a better reconstruction than the baseline (Ours-AD).

4.3 Ablation Study

Differentiable Rendering. To demonstrate the capability of our differentiable renderer, we introduce a challenging scenario by adding zero-mean Gaussian noise to the poses of frames from the N-Matterport3D dataset with a standard deviation of 3cm and 2° for the translation and rotation, respectively. Apart from direct comparisons with the version without differentiable rendering (referring to as "w/o optim."), we verify the effectiveness of our implicit-differentiation-based gradient by replacing it by unrolled iterations obtained through automatic-differentiation [2], denoted by Ours-AD. As Fig. 5 shows, our renderer is able to denoise the input poses, reaching a higher reconstruction quality than its counterparts, the refinements of which are non-trivial due to the discrete TSDF representation used. Moreover, compared to Ours-AD, our full gradient optimization is also more effective, thanks to the accuracy and stability provided by the closed-form derivative computation. Our method also saves a considerable amount of optimization time and memory by avoiding propagating gradients through all points along the ray. A detailed time and memory analysis of our differentiable rendering is given in the supplementary material.

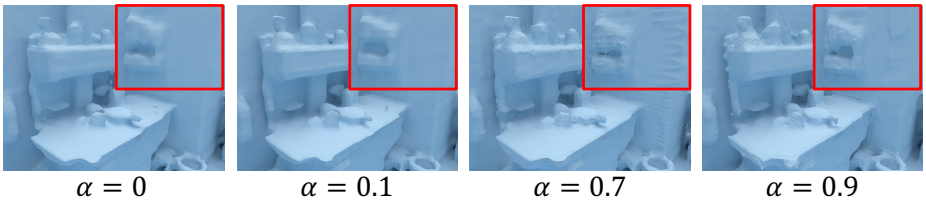

Fig. 6: **Effect of normal loss**. Reconstructions obtained by training with varying weights for $\mathcal{L}_{\text{norm}}$, *i.e.*, α. Red boxes highlight the differences.

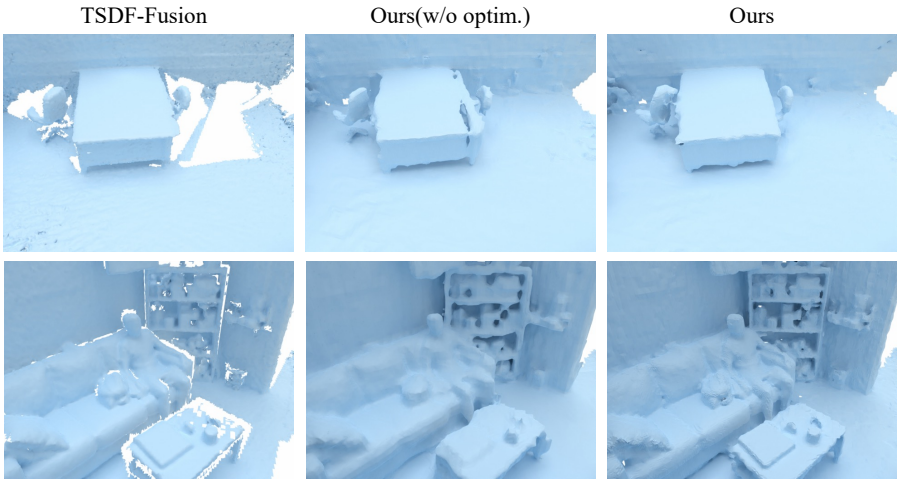

Fig. 7: **Qualitative results using the ScanNetv2 dataset.** Our method generalizes well in presence of real-world noise and occlusions.

Weight of $\mathcal{L}_{\text{norm}}$. After fixing the gauge freedom of the weights for $\mathcal{L}_{\text{sdf}}$ and $\mathcal{L}_{\text{struct}}$ to 1, we show the effect of changing $\mathcal{L}_{\text{norm}}$ in Fig. 6 by varying its weight $\alpha \in [0, 1]$. The addition of normal loss can effectively improve the precision of the reconstruction. However it only works when α is small, showing the importance of carefully choosing the weight parameter, especially in our setting with a small localized voxel size.

Depth Noise. We demonstrate results on the ScanNetv2 dataset in Fig. 7. Although our method is trained using synthetic depth noise, it generalizes well to real-world noise. Compared to the traditional TSDF fusion approach, our differentiable renderer fixes inaccurate poses and significantly sharpens the geometric features. The effect of synthetic noise level on reconstruction results is given in the supplementary material.

Voxel Size. Fig. 8 shows how the reconstruction quality varies when trained and tested with the same voxel size b (using 5cm, 7.5cm and 10cm). A smaller voxel

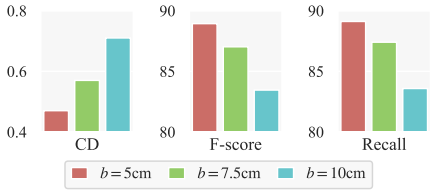

Fig. 8: **Performance for varying voxel sizes.** Our method works best with a small voxel size b.

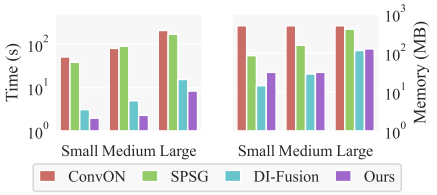

Fig. 9: **Run time and memory.** Results for different methods and different input scene sizes.

size captures more details from the input and models the surface more accurately. It also improves recall by avoiding mis-predicting large regions. Furthermore, we empirically find our method generalizes well across different training voxel sizes: the test error with $b = 7.5$cm is stable (RMSE ≈ 0.019) even if trained using a different voxel size, e.g., 5cm and 10cm. Nevertheless, we recommend a larger voxel size during training to learn more complicated geometries for better generalization.

4.4 Timing and Memory

Due to the differences in scene representation used by each approach, it is hard to fairly compare the timing and memory consumption of the whole pipeline of each method. So we only compare the time to generate the discrete TSDF volume for fairness. In other words, we exclude the time of inference-time refinement and marching cubes, because none of the baselines performs post-optimization and the implementation of marching cubes varies a lot from each other. Because the time comparison between online methods and offline methods is meaningless, we extend DI-Fusion to an offline version and do not consider R-Fusion in following experiment. Fig. 9 compares the inference time and memory footprint of the baselines for different scene sizes. Thanks to the sparse feature volume, our method runs 10–50× faster than ConvON and SPSG, and is comparable in speed to DI-Fusion. However, as the scene gets larger, the time taken by DI-Fusion increases more rapidly than our method due to the difference in voxel interpolation strategy. As for memory cost, ConvON stays constant due to its sliding window inference scheme. SPSG maintains a dense discrete TSDF volume, so memory requirements grow drastically with scene size. Our method is memory-efficient due to its sparse representation and uses only marginally more memory than DI-Fusion while providing better reconstruction accuracy.

As for the time of inference-time refinement, our differentiable renderer could reach an average speed of 1M rays/second (the speed is illustrated in our supplementary material). In our experiments, we render 512×640 pixels for each room. Each room has an average of 44 frames, and we optimize for 20 epochs. In total, it takes ∼440 seconds to optimize one room using PyTorch. This optimizing process can be further speeded up using Jittor [21], a new deep learning

framework which is efficient for training neural networks, and we will also release code using Jittor.

4.5 Limitations and Discussion

Our approach has three main limitations. Firstly, our network makes no use of object-level priors, resulting in partially reconstructed objects even after completion. Training with semantic supervision may improve completion performance. Secondly, reconstruction quality relies on a small voxel size that limits further improvements in efficiency. This can be overcome with local implicit grid [23], which can learn local geometric priors from CAD models using large voxels with further optimization for real-world scenes. Thirdly, textures are not recovered by our method. Inspired by NeRF [33], training a neural radiance field together with SDF using differentiable rendering can incorporate texture information into our pipeline.

5 Conclusions

This paper has introduced CIRCLE, a framework for large-scale scene reconstruction and completion using local implicit signed distance functions. The key part of our method is a convolutional neural network that can learn global contextual information from local implicit grid, contributing to the completion of missing regions. Together with our novel differentiable rendering strategy, we are able to generate an accurate and detailed reconstruction, while being fast and memory-efficient. In the future, we hope to bridge the gap between large-scale geometric reconstruction and the use of object shape priors, as well as to incorporate color information into our pipeline, for better completion and reconstruction.

Acknowledgments. We would like to thank all the anonymous reviewers for their valuable suggestions. This work is supported by the National Key R&D Program of China (Grant No.: 2021ZD0112902) and the National Science Foundation of China (Grant No.: 61902210).

References

1. Azinović, D., Martin-Brualla, R., Goldman, D.B., Nießner, M., Thies, J.: Neural rgb-d surface reconstruction. In: IEEE CVPR. pp. 6290–6301 (2022)
2. Baydin, A.G., Pearlmutter, B.A., Radul, A.A., Siskind, J.M.: Automatic differentiation in machine learning: a survey. Journal of Marchine Learning Research (JMLR) **18**, 153:1–153:43 (2017)
3. Bozic, A., Palafox, P.R., Thies, J., Dai, A., Nießner, M.: Transformerfusion: Monocular RGB scene reconstruction using transformers. In: NeurIPS. pp. 1403–1414 (2021)
4. Cao, Y.P., Kobbelt, L., Hu, S.M.: Real-time high-accuracy three-dimensional reconstruction with consumer RGB-D cameras. ACM Transactions on Graphics (TOG) **37**(5), 171:1–171:16 (2018)

5. Chabra, R., Lenssen, J.E., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., Newcombe, R.A.: Deep local shapes: Learning local SDF priors for detailed 3d reconstruction. In: ECCV. pp. 608–625 (2020)
6. Chang, A.X., Dai, A., Funkhouser, T.A., Halber, M., Nießner, M., Savva, M., Song, S., Zeng, A., Zhang, Y.: Matterport3d: Learning from RGB-D data in indoor environments. In: International Conference on 3D Vision (3DV). pp. 667–676 (2017)
7. Chen, Y., Liu, S., Wang, X.: Learning continuous image representation with local implicit image function. In: IEEE CVPR. pp. 8628–8638 (2021)
8. Chen, Z., Tagliasacchi, A., Zhang, H.: Bsp-net: Generating compact meshes via binary space partitioning. In: IEEE CVPR. pp. 42–51 (2020)
9. Chibane, J., Alldieck, T., Pons-Moll, G.: Implicit functions in feature space for 3d shape reconstruction and completion. In: IEEE CVPR. pp. 6968–6979 (2020)
10. Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: ACM SIGGRAPH. pp. 303–312 (1996)
11. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T.A., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: IEEE CVPR. pp. 2432–2443 (2017)
12. Dai, A., Diller, C., Nießner, M.: SG-NN: sparse generative neural networks for self-supervised scene completion of RGB-D scans. In: IEEE CVPR. pp. 846–855 (2020)
13. Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., Theobalt, C.: Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. ACM Transactions on Graphics (TOG) **36**(3), 24:1–24:18 (2017)
14. Dai, A., Ritchie, D., Bokeloh, M., Reed, S., Sturm, J., Nießner, M.: Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. In: IEEE CVPR. pp. 4578–4587 (2018)
15. Dai, A., Siddiqui, Y., Thies, J., Valentin, J., Nießner, M.: SPSG: self-supervised photometric scene generation from RGB-D scans. In: IEEE CVPR. pp. 1747–1756 (2021)
16. Genova, K., Cole, F., Sud, A., Sarna, A., Funkhouser, T.A.: Local deep implicit functions for 3d shape. In: IEEE CVPR. pp. 4856–4865 (2020)
17. Genova, K., Cole, F., Vlasic, D., Sarna, A., Freeman, W.T., Funkhouser, T.A.: Learning shape templates with structured implicit functions. In: IEEE ICCV. pp. 7153–7163 (2019)
18. Graham, B., Engelcke, M., van der Maaten, L.: 3d semantic segmentation with submanifold sparse convolutional networks. In: IEEE CVPR. pp. 9224–9232 (2018)
19. Handa, A., Whelan, T., McDonald, J., Davison, A.J.: A benchmark for RGB-D visual odometry, 3d reconstruction and SLAM. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 1524–1531 (2014)
20. Hart, J.C.: Sphere tracing: a geometric method for the antialiased ray tracing of implicit surfaces. The Visual Computer **12**(10), 527–545 (1996)
21. Hu, S.M., Liang, D., Yang, G.Y., Yang, G.W., Zhou, W.Y.: Jittor: a novel deep learning framework with meta-operators and unified graph execution. Science China Information Sciences **63**(12), 222103 (2020)
22. Huang, J., Huang, S.S., Song, H., Hu, S.M.: Di-fusion: Online implicit 3d reconstruction with deep priors. In: IEEE CVPR. pp. 8932–8941 (2021)
23. Jiang, C.M., Sud, A., Makadia, A., Huang, J., Nießner, M., Funkhouser, T.A.: Local implicit grid representations for 3d scenes. In: IEEE CVPR. pp. 6000–6009 (2020)
24. Jiang, Y., Ji, D., Han, Z., Zwicker, M.: Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization. In: IEEE CVPR. pp. 1248–1258 (2020)

25. Kato, H., Ushiku, Y., Harada, T.: Neural 3d mesh renderer. In: IEEE CVPR. pp. 3907–3916 (2018)
26. Li, J., Gao, W., Wu, Y., Liu, Y., Shen, Y.: High-quality indoor scene 3d reconstruction with RGB-D cameras: A brief review. Computational Visual Media **8**(3), 369–393 (2022)
27. Liu, L., Gu, J., Lin, K.Z., Chua, T.S., Theobalt, C.: Neural sparse voxel fields. In: NeurIPS. pp. 15651–15663 (2020)
28. Liu, S., Zhang, Y., Peng, S., Shi, B., Pollefeys, M., Cui, Z.: DIST: rendering deep implicit signed distance function with differentiable sphere tracing. In: IEEE CVPR. pp. 2016–2025 (2020)
29. Liu, S., Chen, W., Li, T., Li, H.: Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In: IEEE ICCV. pp. 7707–7716 (2019)
30. Liu, Z.N., Cao, Y.P., Kuang, Z.F., Kobbelt, L., Hu, S.M.: High-quality textured 3d shape reconstruction with cascaded fully convolutional networks. IEEE Transactions on Visualization and Computer Graphics (TVCG) **27**(1), 83–97 (2021)
31. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. In: ACM SIGGRAPH. pp. 163–169 (1987)
32. Mescheder, L.M., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: IEEE CVPR. pp. 4460–4470 (2019)
33. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: representing scenes as neural radiance fields for view synthesis. Communications of the ACM **65**(1), 99–106 (2021)
34. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohli, P., Shotton, J., Hodges, S., Fitzgibbon, A.W.: Kinectfusion: Real-time dense surface mapping and tracking. In: IEEE International Symposium on Mixed and Augmented Reality (ISMAR). pp. 127–136 (2011)
35. Niemeyer, M., Mescheder, L.M., Oechsle, M., Geiger, A.: Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In: IEEE CVPR. pp. 3501–3512 (2020)
36. Oechsle, M., Mescheder, L.M., Niemeyer, M., Strauss, T., Geiger, A.: Texture fields: Learning texture representations in function space. In: IEEE ICCV. pp. 4530–4539 (2019)
37. Oleynikova, H., Taylor, Z., Fehr, M., Siegwart, R., Nieto, J.I.: Voxblox: Incremental 3d euclidean signed distance fields for on-board MAV planning. In: IEEE International Conference on Intelligent Robots and Systems (IROS). pp. 1366–1373 (2017)
38. Park, J.J., Florence, P., Straub, J., Newcombe, R.A., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: IEEE CVPR. pp. 165–174 (2019)
39. Peng, S., Niemeyer, M., Mescheder, L.M., Pollefeys, M., Geiger, A.: Convolutional occupancy networks. In: ECCV. pp. 523–540 (2020)
40. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: IEEE CVPR. pp. 77–85 (2017)
41. Ren, B., Wu, J.C., Lv, Y.L., Cheng, M.M., Lu, S.P.: Geometry-aware icp for scene reconstruction from rgb-d camera. Journal of Computer Science and Technology (JCST) **34**(3), 581–593 (2019)
42. Rosu, R.A., Behnke, S.: Neuralmvs: Bridging multi-view stereo and novel view synthesis. In: IEEE International Joint Conference on Neural Networks (IJCNN) (2022)

43. Sitzmann, V., Martel, J.N.P., Bergman, A.W., Lindell, D.B., Wetzstein, G.: Implicit neural representations with periodic activation functions. In: NeurIPS. pp. 7462–7473 (2020)
44. Song, H., Huang, J., Cao, Y.P., Mu, T.J.: Hdr-net-fusion: Real-time 3d dynamic scene reconstruction with a hierarchical deep reinforcement network. Computational Visual Media **7**(4), 419–435 (2021)
45. Song, S., Zeng, A., Chang, A.X., Savva, M., Savarese, S., Funkhouser, T.A.: Im2pano3d: Extrapolating 360° structure and semantics beyond the field of view. In: IEEE CVPR. pp. 3847–3856 (2018)
46. Sucar, E., Liu, S., Ortiz, J., Davison, A.J.: imap: Implicit mapping and positioning in real-time. In: IEEE ICCV. pp. 6209–6218 (2021)
47. Sun, J., Xie, Y., Chen, L., Zhou, X., Bao, H.: Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In: IEEE CVPR. pp. 15598–15607 (2021)
48. Takikawa, T., Litalien, J., Yin, K., Kreis, K., Loop, C.T., Nowrouzezahrai, D., Jacobson, A., McGuire, M., Fidler, S.: Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In: IEEE CVPR. pp. 11358–11367 (2021)
49. Wang, P.S., Liu, Y., Tong, X.: Deep octree-based cnns with output-guided skip connections for 3d shape and scene completion. In: IEEE CVPR Workshops. pp. 1074–1081 (2020)
50. Weder, S., Schönberger, J.L., Pollefeys, M., Oswald, M.R.: Routedfusion: Learning real-time depth map fusion. In: IEEE CVPR. pp. 4886–4896 (2020)
51. Weder, S., Schönberger, J.L., Pollefeys, M., Oswald, M.R.: Neuralfusion: Online depth fusion in latent space. In: IEEE CVPR. pp. 3162–3172 (2021)
52. Whelan, T., Salas-Moreno, R.F., Glocker, B., Davison, A.J., Leutenegger, S.: Elasticfusion: Real-time dense SLAM and light source estimation. The International Journal of Robotics Research (IJRR) **35**(14), 1697–1716 (2016)
53. Yang, S., Li, B., Cao, Y.P., Fu, H., Lai, Y.K., Kobbelt, L., Hu, S.M.: Noise-resilient reconstruction of panoramas and 3d scenes using robot-mounted unsynchronized commodity RGB-D cameras. ACM Transactions on Graphics (TOG) **39**(5), 152:1–152:15 (2020)
54. Yariv, L., Kasten, Y., Moran, D., Galun, M., Atzmon, M., Basri, R., Lipman, Y.: Multiview neural surface reconstruction by disentangling geometry and appearance. In: NeurIPS. pp. 2492–2502 (2020)
55. Yu, A., Li, R., Tancik, M., Li, H., Ng, R., Kanazawa, A.: Plenoctrees for real-time rendering of neural radiance fields. In: IEEE ICCV. pp. 5732–5741 (2021)