3D-SLNR: A Super Lightweight Neural Representation for Large-scale 3D Mapping

Supplementary Material

In this supplementary material, we first present an overview of our parallel local SDF detection algorithm in Suppl. 1, and then provide additional details regarding the training process in Suppl. 2, followed by the detailed experimental settings in Suppl. 3 and finally, more experimental results in Suppl. 4.

1. Local SDF detection algorithm

Algorithm 1 Parallel local SDF detection for sampled points.

Input: $\boldsymbol{F}^l$: all local SDFs; B: observation range; $\boldsymbol{p}$: sampled points with number n; k: threshold for the number of detected local SDFs.
Output: $H[n][k]$: detected results.

$\boldsymbol{F}^l_c \longleftarrow$ SELECTCANDIDATES$(\boldsymbol{F}^l, B)$
$V \longleftarrow$ BUILDGRID(B)
$key \longleftarrow$ ASIGNKEYS$(\boldsymbol{F}^l_c, V)$
$\boldsymbol{F}^l_{cs} \longleftarrow$ SORTBYKEYS$(\boldsymbol{F}^l_c, key)$
$R \longleftarrow$ INDENTIFYVOXELRANGES$(\boldsymbol{F}^l_{cs}, V)$
for all $\boldsymbol{p}$ in B **do**
  $i_v \longleftarrow$ GETVOXELID$(\boldsymbol{p}, V)$
  $H[i][1:k] \longleftarrow$ DETECTINVOXEL$(\boldsymbol{F}^l_{cs}, V, R, \boldsymbol{p}, i_v)$
end for
return $H[n][k]$

2. Training details

In the training process, we derive the gradients of Eq. 2 in Sec. 3.1 of the main body of the paper and implement them with CUDA to enable more efficient training. The derivations are as follows:

$$\frac{\partial \boldsymbol{p}^l_j}{\partial \mathbf{x}_j} = -\mathbf{S}_j^{-1}\mathbf{R}_j^{-1},$$

$$\frac{\partial \boldsymbol{p}^l_j}{\partial \mathbf{r}_j} = \mathbf{S}_j^{-1}[\mathbf{R}_j^{-1}(\boldsymbol{p}-\mathbf{x}_j)]^{\wedge}\mathbf{J}_l,$$

$$\frac{\partial \boldsymbol{p}^l_j}{\partial \mathbf{s}_j} = -\frac{\mathbf{R}_j^{-1}(\boldsymbol{p}-\mathbf{x}_j)}{exp(\mathbf{s}_j)},$$

$$\frac{\partial \boldsymbol{p}^l_j}{\partial \boldsymbol{p}} = \mathbf{S}_j^{-1}\mathbf{R}_j^{-1},$$

where $\mathbf{S}_j = \begin{bmatrix} exp(\mathbf{s}_{j,1}) & 0 & 0 \\ 0 & exp(\mathbf{s}_{j,2}) & 0 \\ 0 & 0 & exp(\mathbf{s}_{j,3}) \end{bmatrix}$, and $[*]^{\wedge}$ converts a 3D vector into the corresponding antisymmetric matrix. $\mathbf{J}_l$ denotes the left-multiplication BCH approximation Jacobi matrix, which is calculated by:

$$\mathbf{J}_l = \mathbf{I} - \frac{1-cos(\alpha)}{\alpha^2}\mathbf{r}_j^{\wedge} + \frac{\alpha - sin(\alpha)}{\alpha^3}(\mathbf{r}_j^{\wedge})^2,$$

where α is the norm of $\mathbf{r}_j$. Thus, the derivative of SDF value S with respect to the geometric parameters $\{\mathbf{x}_j, \mathbf{r}_j, \mathbf{s}_j\}$ can be easily computed by chain rule:

$$\frac{\partial S}{\partial \mathbf{x}_j} = \frac{\partial S}{\partial \boldsymbol{p}^l_j}\frac{\partial \boldsymbol{p}^l_j}{\partial \mathbf{x}_j}, \frac{\partial S}{\partial \mathbf{r}_j} = \frac{\partial S}{\partial \boldsymbol{p}^l_j}\frac{\partial \boldsymbol{p}^l_j}{\partial \mathbf{r}_j}, \frac{\partial S}{\partial \mathbf{s}_j} = \frac{\partial S}{\partial \boldsymbol{p}^l_j}\frac{\partial \boldsymbol{p}^l_j}{\partial \mathbf{s}_j},$$

where $\frac{\partial S}{\partial \boldsymbol{p}^l_j}$ can be derived automatically by Pytorch. The analytical normal $\mathbf{n}^{\boldsymbol{p}}$ at the sampled point $\boldsymbol{p}$ is:

$$\mathbf{n}^{\boldsymbol{p}} = \nabla_{\boldsymbol{p}} S = \frac{\partial S}{\partial \boldsymbol{p}} = \sum_{j=1}^{K}\frac{\partial S}{\partial \boldsymbol{p}^l_j}\frac{\partial \boldsymbol{p}^l_j}{\partial \boldsymbol{p}},$$

3. Detailed experimental settings

In this section, we describe the detailed experimental settings, including datasets, parameter settings, baselines, and metrics.

3.1. Datasets

We evaluate our approach on the Matterport3D dataset [1], the Newer College dataset [8], a self-collected street-level dataset, and the KITTI-360 dataset [5].

- The Matterport3D dataset is a large-scale indoor RGB-D dataset, where ground truth meshes and frame poses are available. We randomly select 5 scenes for evaluation. Their scene IDs are "1LXtFkjw3qL", "JmbYfDe2QKZ", "ur6pFq6Qu1A", "Vt2qJdWjCF2", and "VzqfbhrpDEA", respectively.
- The Newer College dataset is a hand-carried LiDAR dataset, collected at Oxford University. We evaluate our method in the Quad and Math Institute scenes, where the ground truth point clouds and frame poses are provided.
- Due to the limited availability of outdoor large-scale datasets for 3D mapping evaluation, we construct a street-level dataset. The dataset includes 5 sequences, namely ***real-1***, ***real-2***, ***real-3***, ***real-4***, and ***real-5***, respectively. They are collected by a handheld camera-LiDAR-IMU

integrated device in street scenes. The device is equipped with an IMU, a monocular camera, and a Velodyne VLP-16 LiDAR. The first four sequences are captured along line-shaped streets with lengths of about $230m$, $235m$, $225m$, and $410m$, respectively. The last sequence is collected in a small park with an area of about $370m \times 100m$, and the data collection trajectory is a ring with a length of about $870m$. A NavVis VLX 3 scanner is used to capture the ground truth point cloud of each sequence with millimeter-level accuracy for reconstruction evaluation [1]. Ground truth frame poses are generated by aligning LiDAR frames with the ground truth point clouds. To promote further research in large-scale 3D mapping, we have made the dataset publicly available at the Science Data Bank.

- The KITTI-360 dataset is a kilometer-level dataset, which records several suburbs of Karlsruhe, Germany, corresponding to over 100k laser scans in a driving distance of 73.7km. Accurate vehicle poses are available. Since no ground truth meshes or point clouds are provided, we only perform qualitative experiments on this dataset.

3.2. Parameter settings

Our basis SDF is parameterized by a tiny MLP with 3 layers, each including 64 neural nodes. Its limited radius m is set to 3. In the initialization process, the voxel size s_v for voxel downsampling is set to 0.1, 0.3, 0.5, and 0.5 for the Matterport3D dataset, the Newer College dataset, the street-level dataset, and the KITTI-360 dataset, respectively. In our parallel local SDF detection algorithm, the threshold k for the number of detected local SDFs is set to 12. In the prune-ane-expand step, the distance threshold d_{th} is set to 0.12 and 0.2 for indoor and outdoor scenes, respectively. the gradient threshold x_{th} is set to 2×10^{-4}, and the scale threshold s_{th} is set to $0.8 s_v$. The hyperparameter λ in the loss function was set to 0.02. All experiments are conducted on a desktop with the configuration of Ubuntu 18.04 / Intel Xeon(R)W-2135 CPU / NVIDIA GeForce RTX4090 GPU. For each scene, we train our model using Pytorch for a default of 15000 iterations (about 20 minutes). For the scenes in the KITTI-360 dataset, the model is trained for 50000 iterations. The prune-and-expand strategy is executed after 1000 iterations and every 200 iterations. We adopt the ADAM optimizer for model training. The initial learning rate is 5×10^{-3} and decreases exponentially to 0.1 times the original value.

3.3. Baselines

We compare our approach with a wide range of 3D mapping methods, including VDB-Fusion [9], SPSR [4], NKSR [3], SHINE-Mapping [11], and PIN-SLAM [7]. A brief description of these methods is as follows.

[1] https://www.navvis.com/vlx-3

- VDB-Fusion is a TSDF fusion-based method, which represents a scene as a truncated signed distance function (TSDF) using a discrete VDB [6] grid.
- SPSR fits an indicator field based on B-spline basis functions anchored at nodes of an octree.
- NKSR models scenes using neural kernels constructed from learnable latent features. In our experiments, we adopt the official kitchen-sink-model.
- SHINE-Mapping implicitly represents a scene as an SDF using a hierarchical feature grid and a small multi-layer perceptron (MLP). In our experiments, we perform the batch mapping mode.
- PIN-SLAM models scenes with a set of neural points and an MLP decoder. The approach is a simultaneous localization and mapping (SLAM) system. In our experiments, we deactivate its pose estimation module and use ground truth poses for incremental mapping. We also attempted to adapt PIN-SLAM for batch mapping, but it was unsuccessful. The issue arises in large-scale scenes, where its hash indexing results in numerous conflicts, leading to degraded performance.

For SPSR and NKSR, we estimate point normals by performing the principal component analysis (PCA) on neighbor points and then correcting these normals toward the data acquisition sensor, the same as the procedure used for rotation parameter initialization of local SDFs, described in Sec. 3.1 of the main body of the paper.

3.4. Metrics

We adopt the commonly used metrics named accuracy (Acc.) in centimeter (cm), completeness (Comp.) in cm, precision (Prec.) in percent (%), recall in %, and F-score in %, where Prec., recall and F-score are presented with a $10cm$ error threshold in indoor scenes and a $20cm$ error threshold in outdoor scenes. We also evaluate the memory consumption for map representation (MC) in megabytes (M). All data stored in memory is counted as the float type. Below, we outline the methodology for calculating the memory consumption for map representations across different methods.

- VDB-Fusion: we account for the number of nodes of the VDB data structure, where each node is viewed as a float-type value. We also consider the SDF values and their associated weights stored in the leaf nodes.
- SPSR: we count the number of octree nodes, each represented as float-type data, and account for the weights of the B-spline basis functions stored in the active multi-level nodes.
- NKSR: we consider the latent features generated during the reconstruction process, including structure features, normal features, kernel features, and mask features.
- SHINE-Mapping: we consider the memory usage of the hierarchical feature grid and the MLP.

- PIN-SLAM: we account for the latent features associated with neural points and their geometric parameters (position and rotation), as well as the MLP decoder.
- Ours: we evaluate the memory consumption of the geometric parameters of local SDFs and the tiny MLP representing the SDF basis.

4. More experimental results

4.1. Ablation study

Local SDF. We compare with some other forms of local SDF [2, 10]. Let's ignore the latent features. Then, the formulas of [2, 10] and ours can be generally written as: $sdf(\boldsymbol{x}) = \sum_{i\in[N]} \varphi_i(\boldsymbol{x}) f_{\boldsymbol{\phi}}(T_i(\boldsymbol{x}))$, where $\boldsymbol{x}$ is a sampled point in the world coordinate system, $T_i(\boldsymbol{x})$ is a transformation of $\boldsymbol{x}$ into the local coordinate system, $\varphi_i(\boldsymbol{x})$ is a weighting function, $f_{\boldsymbol{\phi}}(T_i(\boldsymbol{x}))$ is a local SDF, and $f_{\boldsymbol{\phi}}(T_i)$ is an MLP-parameterized local basis SDF with the parameters $\boldsymbol{\phi}$. The novelty of our representation lies in a different $T_i(\boldsymbol{x})$. In [2], $T_i^1(\boldsymbol{x}) = \boldsymbol{x}$ (*Conf.1*), and in [10], $T_i^2(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{\mu_i}$ (*Conf.2*), where $\boldsymbol{\mu}_i$ can be viewed as the position of the local SDF. In contrast, we also consider the rotation $\boldsymbol{R}_i$ and the scaling $\boldsymbol{S}_i$ of the local SDF, and $T_i^3(\boldsymbol{x}) = \boldsymbol{R}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu_i})\boldsymbol{S}_i^{-1}$ (*Conf.3*), which greatly eliminates the dependence on local latent features. Additionally, we compare with the configuration of "*Config3+Latent features*", which can be formulated to $sdf(\boldsymbol{x}) = \sum_{i\in[N]} \varphi_i(\boldsymbol{x}) f_{\boldsymbol{\phi}}(T_i^3(\boldsymbol{x}), h_i)$ with h_i denoting each latent feature. The reconstruction results are shown in Fig. 1. Compared to "*Config.1*" and "*Config.2*", our method significantly improves the reconstruction performance. When compared to "*Config3+Latent features*", ours achieves comparable reconstruction performance with less memory usage.

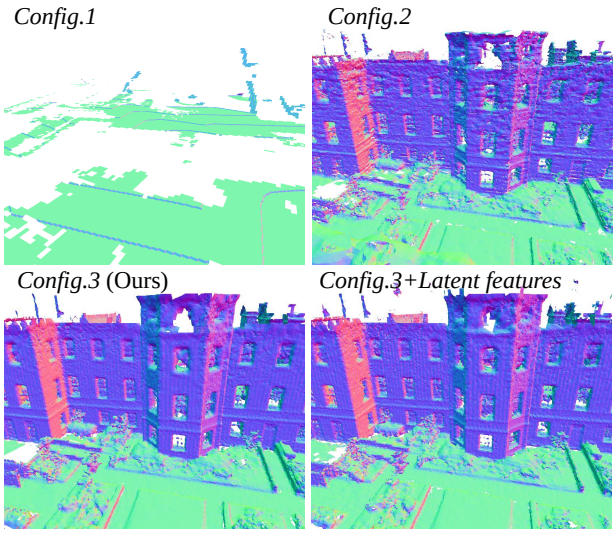


Figure 1. Reconstruction results for different forms of local SDF in the Math Institute scene.

Voxel downsampling. This paper adopts uniform voxel downsampling to initialize support points. This approach is easy to implement and efficient, aiding downstream tasks that require high efficiency. We experiment with non-uniform downsampling in the Math Institute scene. Specifically, we implement a hybrid voxel data structure that allocates smaller voxels in geometrically complex areas and larger voxels in geometrically simple areas. The results, shown in Tab. 1, indicate a decrease in accuracy for 'non-uniform'. We believe the main reason is that geometrically complex areas can contain substantial noise (e.g., dynamic objects), and allocating more support points in these areas during the initial training stages may lead to overfitting the noise.

voxel downsampling	Prec.	Recall	F-Score	MC
uniform	82.4	88.5	85.3	8.0
non-uniform	80.6	88.8	84.5	8.0

Table 1. Experimental results of voxel downsampling in the Math Institute scene.

Thresholds of prune-and-expand strategy. We do some experiments in the Math Institute scene. The results are listed in Tab. 2. The default parameters in the paper are in bold. For the experiments of each threshold, other thresholds remain default. We can see that the default configuration provides a good balance between reconstruction quality and memory usage.

d_{th}	F-score	MC	x_{th}	F-score	MC	s_{th}	F-score	MC
0.05	84.6	2.1	5×10^{-5}	85.3	10.0	0.06	85.3	8.1
0.1	85.1	4.6	1×10^{-4}	85.1	8.2	0.15	85.3	8.1
0.2	85.3	8.0	$\mathbf{2 \times 10^{-4}}$	85.3	8.0	**0.24**	85.3	8.0
0.3	85.3	9.4	4×10^{-4}	85.3	8.0	0.36	85.3	8.1

Table 2. Experimental results for different thresholds of the prune-and-expand strategy in the Math Institute scene.

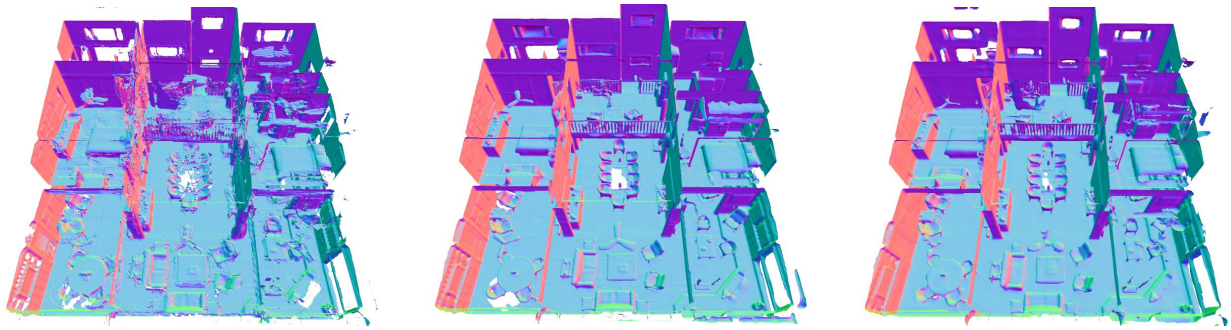
F-score trend for the voxel size s_v. We do some tests for more parameter settings to show the trend of the F-score, which initially increases and then decreases (Tab. 3).

s_v (m)	2.0	1.5	1.0	0.5	0.3	0.2	0.1
F-score	88.4	90.5	91.7	93.8	92.7	92.1	90.1

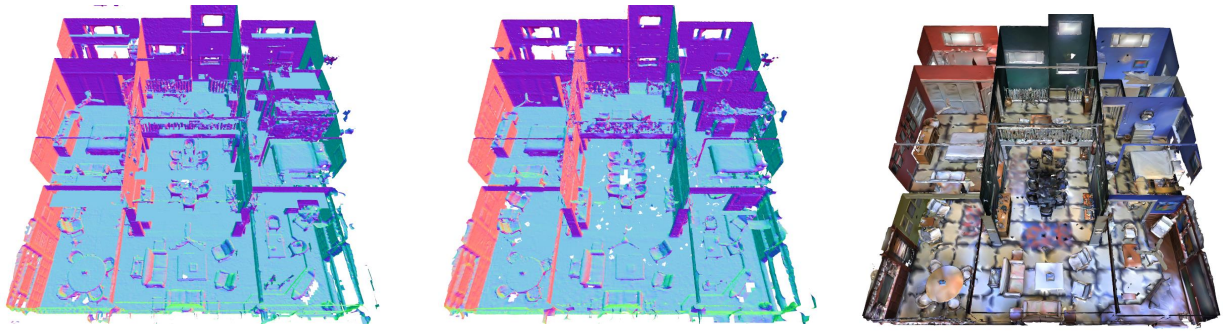
Table 3. The reconstruction results with respect to s_v.

4.2. Results of comparative experiments

We present more experimental results evaluated on the Matterport3D dataset, the street-level dataset, and the KITTI-360 dataset in Fig. 2, Fig. 3, Fig. 4, Fig. 5, Fig. 6, Fig. 7, Fig. 8, Fig. 9, and Fig. 10. These results provide

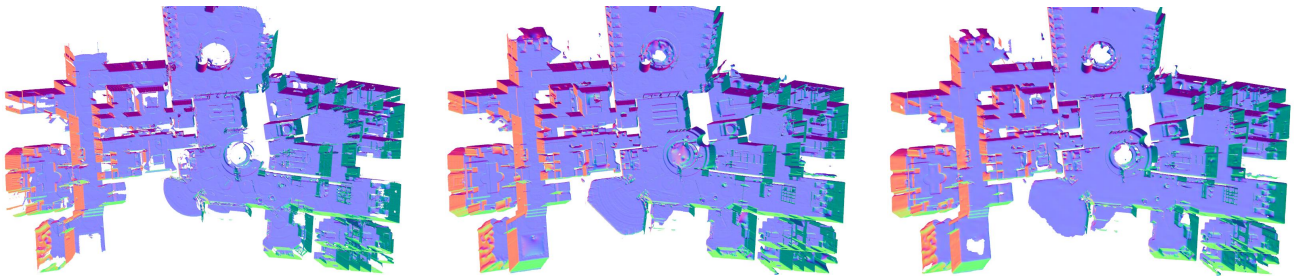


VDB-Fusion (F-score: 91.1, MC: 95.8) SPSR (F-score: 97.0, MC: 25.2) NKSR (F-score: 96.9, MC: 103.8)

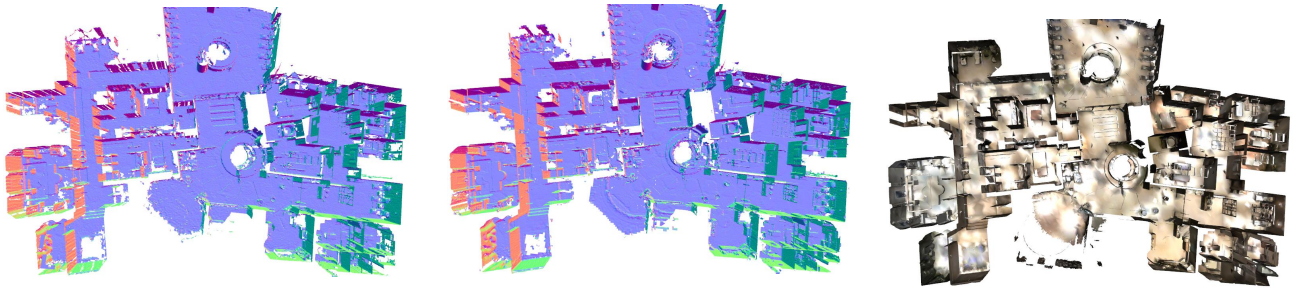


SHINE-Mapping (F-score: 94.2, MC: 72.2) **Ours** (F-score: 97.0, MC: 5.2) ground truth mesh

Figure 2. Experimental results evaluated in "JmbYfDe2QKZ" (about $14.9m \times 16.8m \times 6.4m$) in the Matterport3D dataset. The reconstructed meshes are colored by surface normals.



VDB-Fusion (F-score: 100.0, MC: 93.0) SPSR (F-score: 94.2, MC: 44.7) NKSR (F-score: 94.4, MC: 115.4)



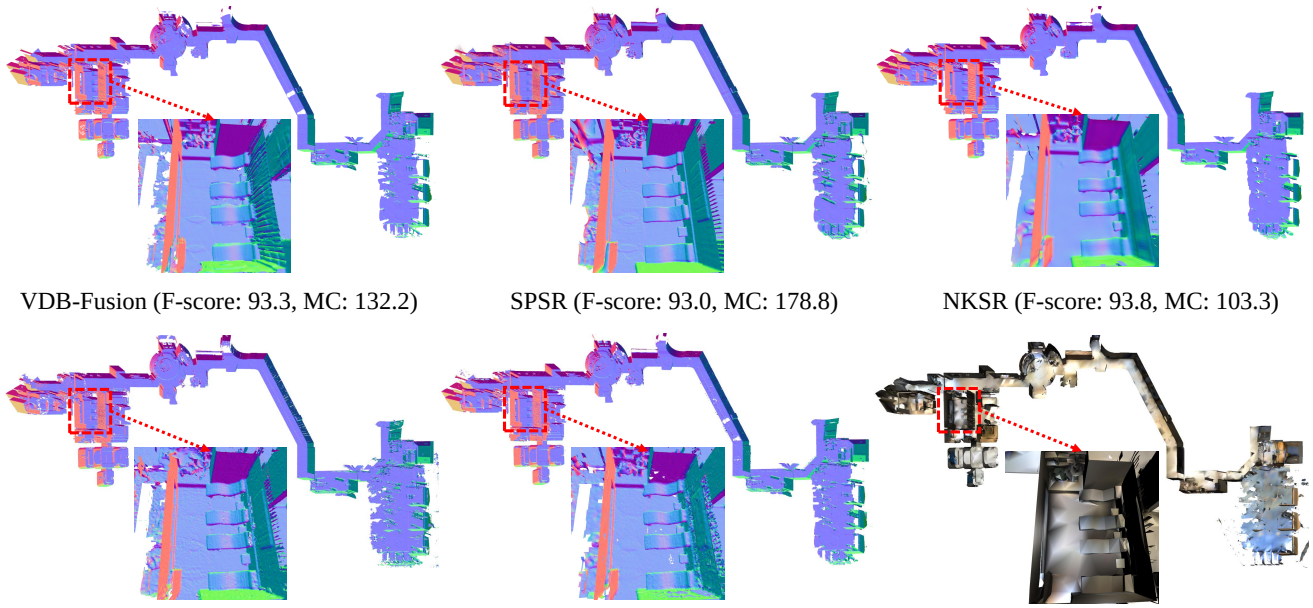
SHINE-Mapping (F-score: 92.4, MC: 136.8) **Ours** (F-score: 94.5, MC: 18.3) ground truth mesh

Figure 3. Experimental results evaluated in "ur6pFq6Qu1A" (about $33.8m \times 44.5m \times 3.6m$) in the Matterport3D dataset. The reconstructed meshes are colored by surface normals.

ample evidence for the extreme compactness and excellent expressiveness of our 3D representation.

References

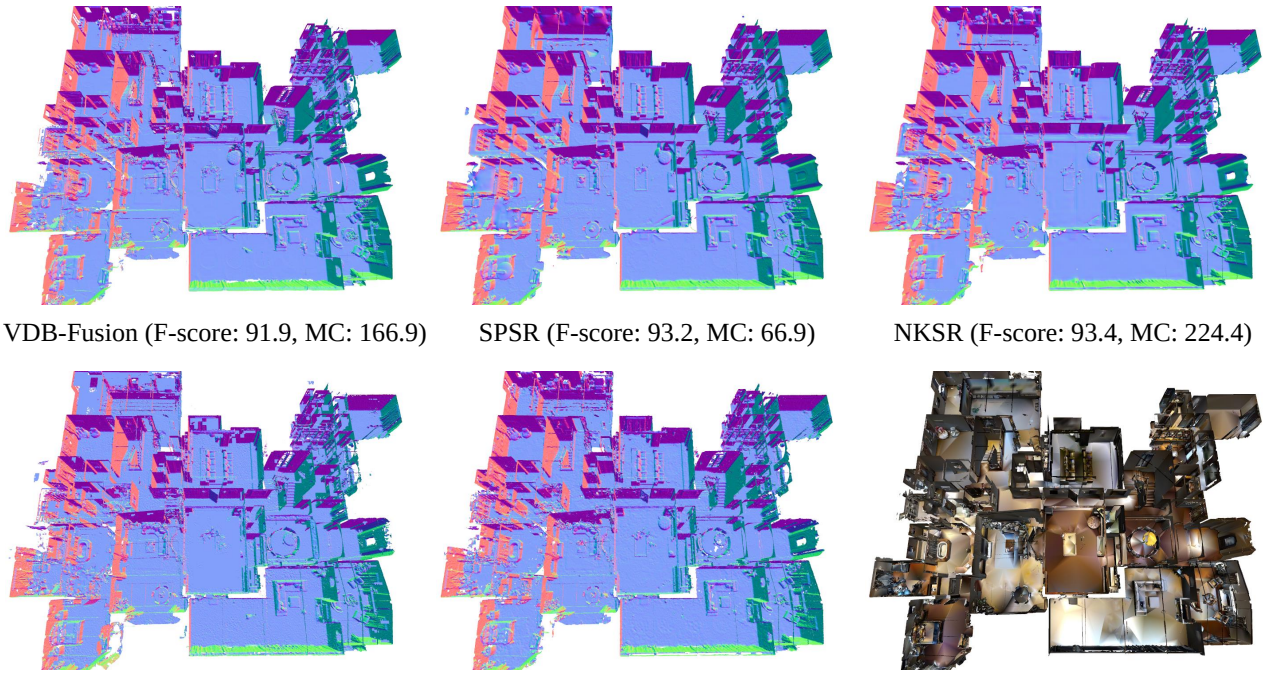
[1] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song,



VDB-Fusion (F-score: 93.3, MC: 132.2) SPSR (F-score: 93.0, MC: 178.8) NKSR (F-score: 93.8, MC: 103.3)

SHINE-Mapping (F-score: 92.0, MC: 140.2) **Ours** (F-score: 93.8, MC: 9.9) ground truth mesh

Figure 4. Experimental results evaluated in "Vt2qJdWjCF2" (about $100m$ long) in the Matterport3D dataset. The reconstructed meshes are colored by surface normals.



VDB-Fusion (F-score: 91.9, MC: 166.9) SPSR (F-score: 93.2, MC: 66.9) NKSR (F-score: 93.4, MC: 224.4)

SHINE-Mapping (F-score: 92.8, MC: 342.2) **Ours** (F-score: 93.7, MC: 24.2) ground truth mesh

Figure 5. Experimental results evaluated in "VzqfbhrpDEA" (about $41.4m \times 34.6m \times 7.1m$) in the Matterport3D dataset. The reconstructed meshes are colored by surface normals.

Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017. 1

[2] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *Proceedings of the IEEE/CVF conference on*

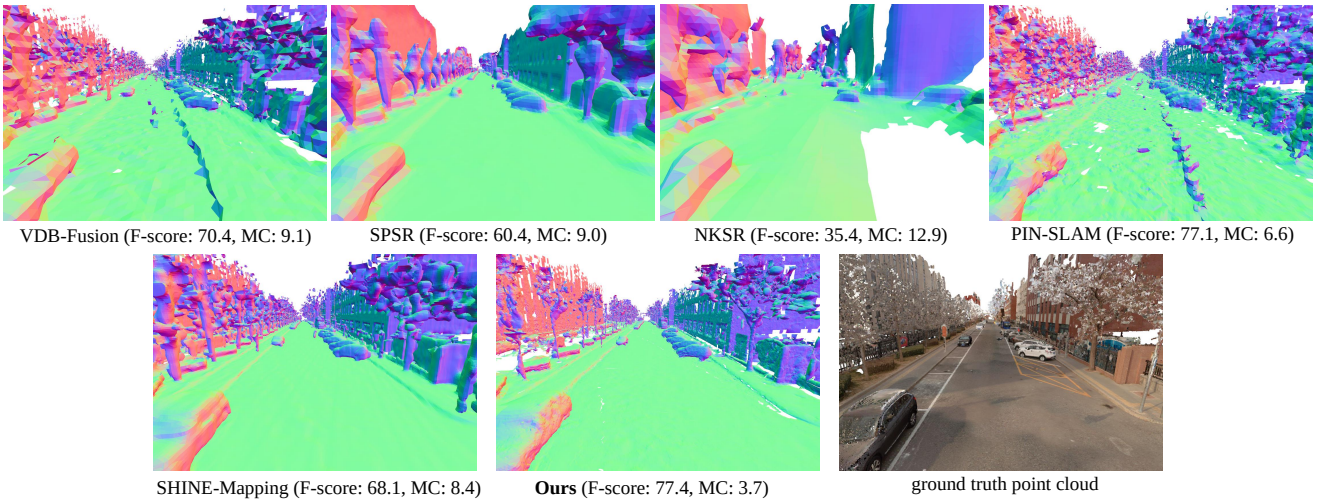


Figure 6. Experimental results evaluated in ***real-1*** (about $230m$ long) in the street-level dataset.

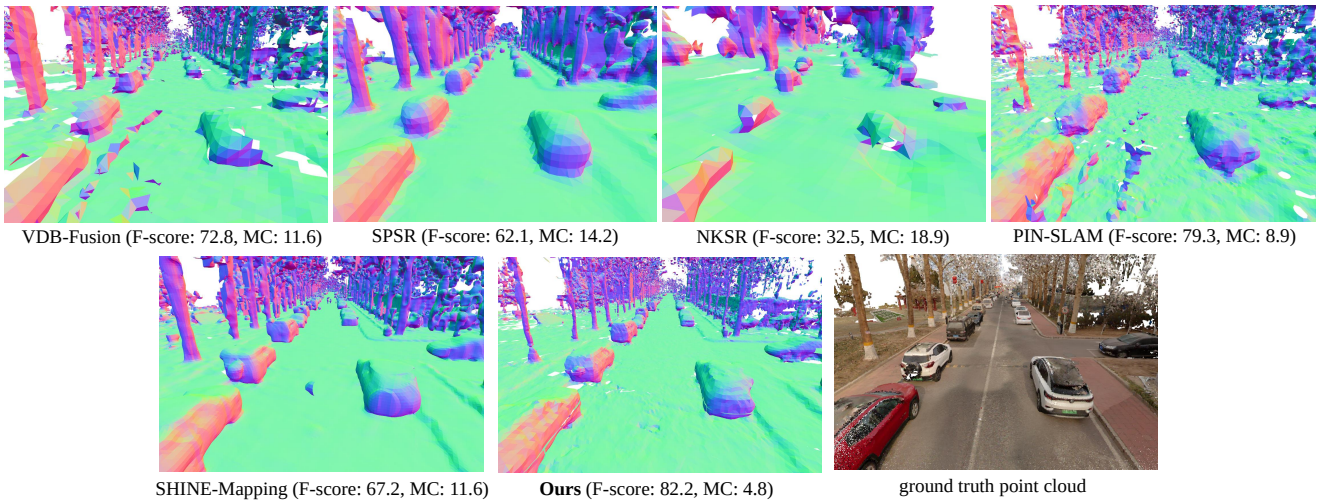


Figure 7. Experimental results evaluated in ***real-3*** (about $225m$ long) in the street-level dataset.

computer vision and pattern recognition, pages 4857–4866, 2020. 3

[3] Jiahui Huang, Zan Gojcic, Matan Atzmon, Or Litany, Sanja Fidler, and Francis Williams. Neural kernel surface reconstruction. In *Proceedings of the IEEE/CVF Conference on CVPR*, pages 4369–4379, 2023. 2

[4] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013. 2

[5] Yiyi Liao, Jun Xie, and Andreas Geiger. Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3292–3310, 2022. 1

[6] Ken Museth. Vdb: High-resolution sparse volumes with dynamic topology. *ACM transactions on graphics (TOG)*, 32 (3):1–22, 2013. 2

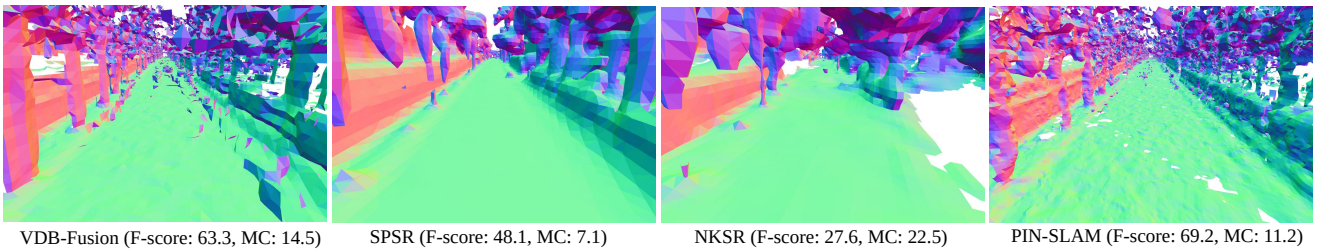
[7] Yue Pan, Xingguang Zhong, Louis Wiesmann, Thorbjörn Posewsky, Jens Behley, and Cyrill Stachniss. Pin-slam: Lidar slam using a point-based implicit neural representation for achieving global map consistency. *arXiv preprint arXiv:2401.09101*, 2024. 2

[8] Milad Ramezani, Yiduo Wang, Marco Camurri, David Wisth, Matias Mattamala, and Maurice Fallon. The newer college dataset: Handheld lidar, inertial and vision with ground truth. In *2020 IEEE/RSJ International Conference on IROS*, pages 4353–4360. IEEE, 2020. 1

[9] Ignacio Vizzo, Tiziano Guadagnino, Jens Behley, and Cyrill Stachniss. Vdbfusion: Flexible and efficient tsdf integration of range sensor data. *Sensors*, 22(3):1296, 2022. 2

[10] Hui Ying, Tianjia Shao, He Wang, Yin Yang, and Kun Zhou. Adaptive local basis functions for shape completion. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023. 3

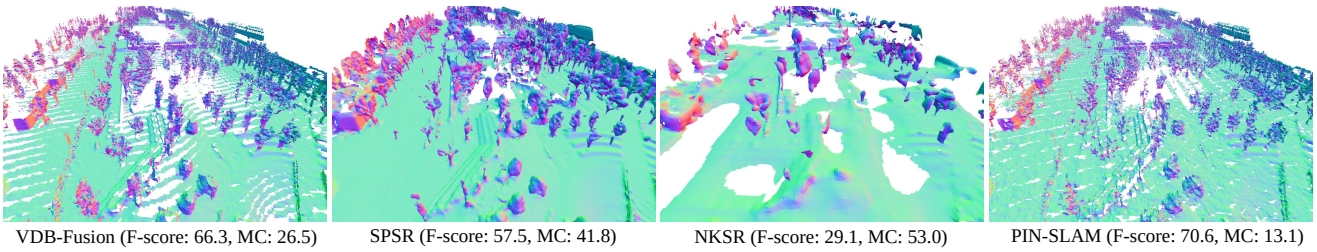
[11] Xingguang Zhong, Yue Pan, Jens Behley, and Cyrill Stach-



VDB-Fusion (F-score: 63.3, MC: 14.5) SPSR (F-score: 48.1, MC: 7.1) NKSR (F-score: 27.6, MC: 22.5) PIN-SLAM (F-score: 69.2, MC: 11.2)

SHINE-Mapping (F-score: 60.0, MC: 13.8) **Ours** (F-score: 72.0, MC: 6.0) ground truth point cloud

Figure 8. Experimental results evaluated in ***real-4*** (about $410m$ long) in the street-level dataset.



VDB-Fusion (F-score: 66.3, MC: 26.5) SPSR (F-score: 57.5, MC: 41.8) NKSR (F-score: 29.1, MC: 53.0) PIN-SLAM (F-score: 70.6, MC: 13.1)

SHINE-Mapping (F-score: 57.7, MC: 23.9) **Ours** (F-score: 72.2, MC: 12.6) ground truth point cloud

Figure 9. Experimental results evaluated in ***real-5*** (about $370m \times 100m$) in the street-level dataset.

niss. Shine-mapping: Large-scale 3d mapping using sparse hierarchical implicit neural representations. In *2023 IEEE ICRA*, pages 8371–8377. IEEE, 2023. 2

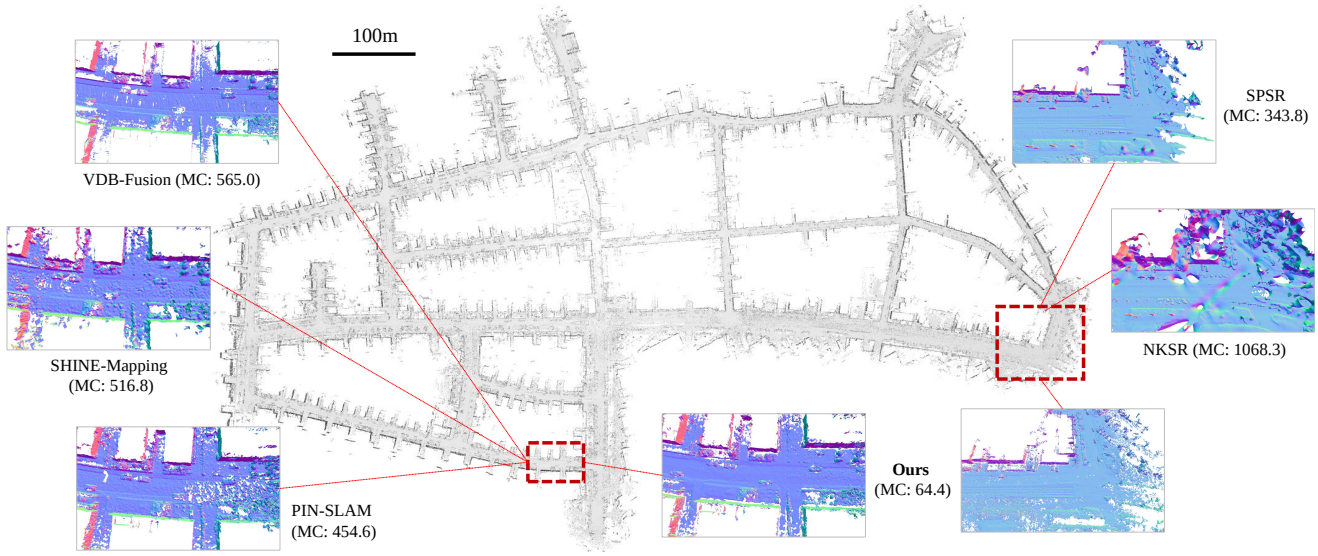


Figure 10. Experimental results evaluated in the KITTI-360 dataset. Our approach reconstructs more consistent and smoother meshes in a kilometer-scale scene with less than 1/5 memory consumption for map representation, compared to previous advanced methods.