NeRF-based Localization and Meshing with Wearable Laser Scanning System: A Case Study in Underground Environment

Yizhe Zhang[1], Jianping Li [2], Xin Zhao[1], Youqi Liao[1], Zhen Dong[1], Bisheng Yang[1]

[1] LIESMARS, Wuhan University, Wuhan, China, 430079 - yizhezhang0418@163.com, 1056810788@qq.com, martin_liao@whu.edu.cn, dongzhenwhu@whu.edu.cn, bshyang@whu.edu.cn
[2] Nanyang Technological University, 50 Nanyang Avenue, Singapore, 639798 - lijianping@whu.edu.cn

Keywords: NeRF, SLAM, Underground, Meshing, Wearable Sensing

Abstract

Due to the subterranean scene’s poor lighting conditions and variability of the environment, real-time localizing and meshing in complex underground scenes present a challenging task, with high-potential applications in mining and tunnel protection. In this work, we propose a method that combines SLAM and NeRF for mesh reconstruction in the underground environment with a wearable device. First, a LiDAR-Inertial odometry is used for pose estimation. The resulting poses and sequential laser frames are synchronized to generate a novel data structure, scan-block, which is crucial for improving efficiency and ensuring local precision. This structure is designed to enhance efficiency and ensure local precision. Finally, the generated scan-blocks are passed to the back-end NeRF for real-time mesh reconstruction. The experiments are carried out in a complex underground space. The experimental results proved that this method achieved good results. The demo video can be found at `https://youtu.be/_y1vbaW06EM?si=hFnAl12u-ffU933h`.

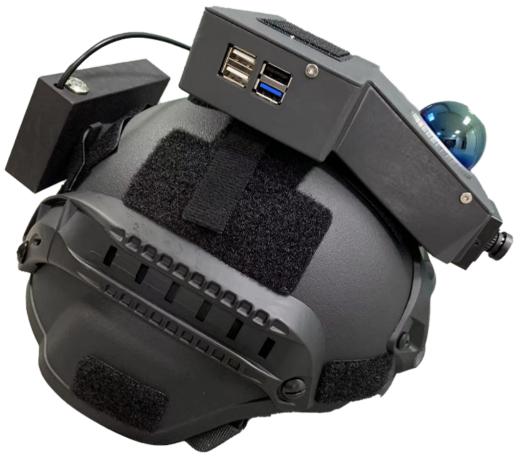

Figure 1. Helmet equipped with MID-360

1. Introduction

Mesh is a graphical object composed of vertices, edges, and faces, utilized to depict the geometric structure of a 3D graph. It holds a central position in the field of 3D reconstruction due to its capability to accurately represent complex shapes and scenes. Mesh reconstruction in underground spaces is a pivotal technological step, playing a significant role across various fields. It enables precise depiction of the shape and positioning of underground pipelines, furnishing a database for subsequent analysis, design, and management (Franzius and Potts, 2005), thereby contributing to the development of smart cities (Yang et al., 2023). Furthermore, in geological exploration and mineral extraction processes, precise underground 3D models aid engineers in comprehending underground structures and anticipating potential issues, thereby mitigating risks and enhancing efficiency and safety (Long et al., 2018). Nevertheless, real-time mesh reconstruction encounters numerous challenges in underground environments, rendering it a formidable task.

The first challenge in real-time meshing lies in the complexity of terrain and structures. Underground spaces often feature intricate landscapes and man-made structures like tunnels, subway stations, pipelines, and caves. These structures vary in shape and size and may exhibit complex connections and intersections, posing a significant hurdle for accurate modeling (Hashimoto and Saito, 2019). Another obstacle is the difficulty of data collection. Acquiring data in subsurface environments necessitates specialized equipment and techniques such as subsurface scanning radar (GPR), laser scanning, and optical measurement techniques (Park et al., 2017). These methods may encounter various issues underground, including signal attenuation, noise interference, and insufficient light, all of which can compromise the accuracy and completeness of the data (Li et al., 2024, Li et al., 2022). Moreover, real-time data processing methods and mesh construction present additional challenges. Transitioning from raw data to the final mesh model entails multiple steps, such as feature extraction, reconstruction, and optimization (Remondino, 2003). These processes involve intricate algorithms that demand substantial computational resources and impose stringent requirements on algorithm efficiency and accuracy.

SLAM (Simultaneous Localization And Mapping, SLAM) is an essential method for real-time meshing, and in recent years, it has made considerable advancements, becoming relatively mature in certain domains. However, many experiments and applications now occur in urban or indoor settings, which introduces limitations. In particularly challenging environments like underground scenes, traditional SLAM methods face significant hurdles. To address these challenges, researchers have explored the use of a combined LiDAR-camera-IMU approach for reconstructing underground scenes. While some progress has been made, obstacles persist due to the complex terrain, sparse point cloud data, and poor illumination conditions inherent to underground environments. Furthermore, there has been limited focus on leveraging SLAM techniques specifically for mesh reconstruction, with only a few researchers exploring this

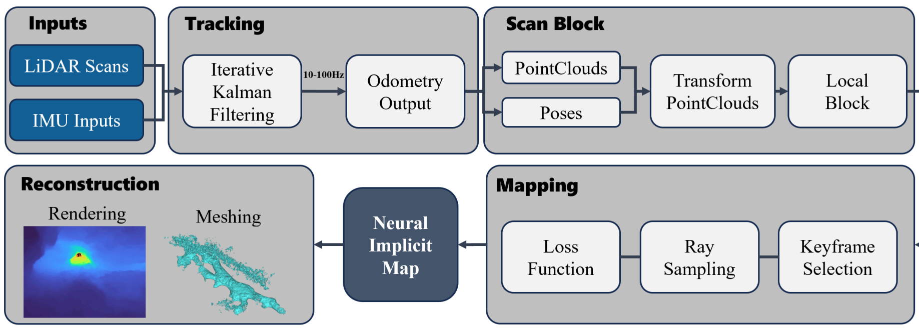

Figure 2. Pipeline of our algorithm. It consists of three main sections. The final implicit map can be rendered as a mesh.

area (Ruan et al., 2023).

NeRF (Neural Radiance Field) proposed in recent years can reconstruct high-quality 3D scenes and render from sparse pictures, thus solving the problem of poor reconstruction effect due to sparse data. iMAP (Sucar et al., 2021) was the first to combine SLAM with NeRF, and NICE-SLAM (Zhu et al., 2022) was modified from iMAP to significantly improve the accuracy and efficiency of the optimization. NeRF-LOAM (Deng et al., 2023) used the LiDAR data to construct a comprehensive 3D representation of large-scale environments, solving the problem of building maps with sparse but accurate data. LONER (Isaacson et al., 2023) is the first real-time NeRF-based LiDAR SLAM. However, NeRF-based LiDAR SLAM is computationally intensive, and its current front-end pose estimation is overly simplistic. As a result, it struggles to match the positioning and reconstruction accuracy of existing traditional SLAM methods.

Building upon these challenges, this paper proposes an approach to mesh reconstruction by integrating NeRF with SLAM and evaluating the complex underground environment. The front end of the system employs traditional SLAM techniques to compute poses. The resulting poses and LiDAR frames are then used to generate scan-blocks and forwarded to the back-end for NeRF training and mesh reconstruction. To optimize computational resources and ensure efficiency, keyframes for back-end construction are selected based on temporal criteria.

The primary contributions of this paper encompass the following aspects:

1) Compared to most SLAM systems that use the least square method for front-end pose estimation, we employ the more robust extended Kalman filter for pose estimation to improve system reliability.

2) We use a novel data structure, the scan block, as a way to guarantee local geometric accuracy while boosting the real-time performance of the system.

3) We use a real-time NeRF-based SLAM system for mesh reconstruction of underground scenes and conduct experiments in several environments to validate the effectiveness of our proposed method.

The remainder of this paper is organized as follows. The wearable sensing system is briefly introduced in 2. The method is revealed in Section 3. In Section 4, the experimental studies are undertaken to evaluate the proposed method, after which conclusions are drawn in Section 5.

2. Sensors and Configuration

Our wearable sensing system comprises a MID-360 LIDAR with its integrated IMU, housed within a helmet featuring signal transceiver functionality, as illustrated in Fig. 1. The hardware design of the helmet aligns with that of the WHU-helmet (Li et al., 2023), with detailed specifications outlined in the referenced paper. During data collection, the LIDAR measures the data while the IMU gathers inertial measurements. Subsequently, the collected information is transmitted to a miniature laptop via a signal transmitter.

The MID-360 LiDAR distinguishes itself from traditional mechanical LiDAR by offering a horizontal field of view of 360° and a vertical field of view of 59°. This wider field of view enables it to capture more comprehensive point cloud data for measurements. Moreover, the helmet is lightweight, facilitating prolonged portability for underground workers during extended periods of work.

3. Method

3.1 System Overview

The workflow of the system is depicted in Fig.2. Similar to typical SLAM systems, this system comprises two main parallel structures: front-end tracking and back-end mapping. The front-end thread handles incoming LiDAR scans and IMU measurements, with state estimation performed by an iterative Kalman filter (Xu et al., 2022). Subsequently, the estimated pose and point cloud are jointly processed to create a local data block. Concurrently, at a lower frequency, the mapping thread updates the trained neural scene representation using the current data block and selects previous data blocks as keyframes.

3.2 Tracking

The tracking thread receives the data emitted by the helmet and passes it into the front-end odometry. The raw LiDAR points sampled sequentially are first accumulated in certain time periods. The accumulated point cloud is called a scan. Since LiDAR usually collects point one by one, the resulting points are sampled at different poses during the continuous motion of the LiDAR. To correct for this in-scan motion, we use the method in FAST-LIO (Xu and Zhang, 2021) to back-estimate the LiDAR position of each point in the scan based on the position of the IMU pre-integrated value with respect to the time of the end of the scan thus back-estimating the LiDAR position for each point in the scan, thus projecting all the scanned points to the end moment based on the exact sampling time of each point in the

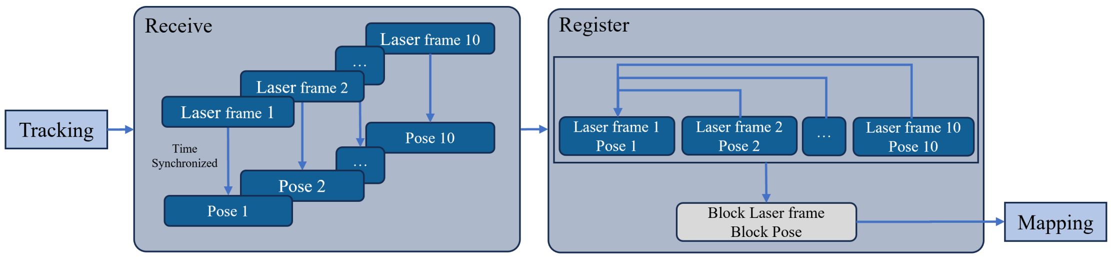

Figure 3. Generation of scan block. This picture clearly gives the flow of forming a scan block. Multiple laser scans and poses are synchronized in time, and the frames of the block are aligned to the first frame by pose to form a local block.

scan. Thus, the projected scan points can be viewed as sampled simultaneously at the end time of the scan.

Due to the presence of LiDAR measurement noise ${}^{L}n_j$, each measurement point is usually polluted by measurement noise, and removal of this noise yields the true point position ${}^{L}p_j$ in the local LiDAR coordinate system. At moment k, this real point should lie exactly in the small local plane under the projection to the global coordinate system using the corresponding LiDAR position ${}^{G}T_{I_k} = \left({}^{G}R_{I_k}, {}^{G}p_{I_k}\right)$ and the extristric parameter ${}^{G}T_L$, i.e., the measurement model is:

$$0 = {}^{G}u_j^T \left({}^{G}T_{I_k}^{I} T_L \left({}^{L}p_j + {}^{L}n_j - {}^{G}q_j\right)\right. \tag{1}$$

where ${}^{G}u_j$ is the normal vector in the corresponding plane and ${}^{G}q_j$ is a point located in the plane. Both ${}^{G}T_{I_k}$ and ${}^{I}T_L$ are contained in the state vector x_k.

We use an iterative Kalman filter for state estimation, consisting of two key steps: propagation of each IMU measurement and iterative updating for each LiDAR scan. Typically, since IMU measurements are more frequent than LiDAR, multiple propagation steps are required before the update. The forward propagation takes place when the IMU measurements arrive and continues throughout. After the current state is obtained through forward propagation, residual computation is performed through the measurement equations, and finally an iterative update is performed to obtain the current state estimate $\hat{x}_k$ and the point cloud $\left\{{}^{G}\overline{p}_j\right\}$.

3.3 Scan Block

We divide the point clouds measured by the Wearable Laser Scanning System into several groups of scan blocks based on time intervals and trajectories (Yang and Li, 2022), as shown in Fig. 3. Each scan block receives poses estimates from tracking $T_{i,j}(j = 1, 2, \ldots, 10)$ and a single-frame point cloud $frame_{i,j}$, and performs the time synchronization to form a local scan block, denoted as SC_i.

In this SC_i, we use the point cloud of the first frame $frame_{i,1}$ as the reference point cloud and the pose of the first frame $T_{i,1}$ as the reference pose. Since we already know the pose $T_{i,j}$ and point cloud $frame_{i,j}$ of each frame estimated by tracking, we can use Eq:

$$frame_i' = frame_{i,1} \oplus (T_{i,1}^{-1} * T_{i,j} * frame_{i,j}), j = 2, 3, \ldots, 10 \tag{2}$$

to transform other frames to the first frame. The notation $\oplus$ denotes the stitching point cloud operation.

Simplify the processed localized scan block as:

$$SC_i = ((frame_i', T_{i,1}), \ldots)$$

The scan block is updated and sent to the mapping thread at 1Hz for NeRF training.

3.4 Implicit Map Representation

We used the method mentioned in (Isaacson et al., 2023) to represent the scene with an MLP containing a hierarchical feature grid. With online training, the parameters of the MLP and the feature grid are synchronously updated to predict the volume density σ of each point in space. For a LiDAR ray $\vec{r}$ with an origin $\vec{o}$ and direction vectors $\vec{d}$, we select N samples along the ray. Occupancy state σ_i will be predicted by feature gird and trained MLP. Cumulative transmittance T_i and weight w_i are calculated by the following equations:

$$T_i = exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j) \tag{3}$$

$$w_i = T_i \sigma_i \tag{4}$$

where δ_j represents the distance between two sampling points. The weights w_i represent the probability of where each line will end up along the direction vector. Therefore, the depth along each line can be predicted as:

$$\hat{ED}(\vec{r}) = \sum_{i=1}^{N} w_i t_i \tag{5}$$

3.5 Mapping

The mapping thread receives the scan block from the tracking thread and decides whether to form a keyframe. If the scan block is successfully received, the map will be co-optimized with the pose.

3.5.1 Keyframe Keyframes are selected based on a temporal criterion. Whenever a certain time interval t has elapsed, a new keyframe is added to the mapping thread. During each optimization update, the current keyframe and $N_{KF} - 1$ randomly selected past keyframes are jointly optimized.

3.5.2 Optimization When a window for the optimized keyframes is selected, the map and the keyframes' poses will be jointly optimized within the optimization window. For a keyframe with an estimated pose $\hat{x}_k$, a vector $\hat{\delta}_i \in R^6$ is used as optimization variable. In the forward propagation stage, this vector is converted back to a pose $\hat{x}_i$ and used to compute the origin of the ray. In the back propagation stage, the gradient will be computed for the MLP from the vector $\hat{\delta}_i$ along with the parameters of the MLP. The final optimized vector will be transformed into a pose x_i^*. At the same time, the optimization will be synchronized to the tracking thread, so that subsequent tracking is based on the optimized poses.

3.6 Loss Function

There are two terms that comprise our loss function, the JS loss and the depth loss. The loss function is as

$$L(\theta) = L_{JS} + \lambda_1 L_{depth} \quad (6)$$

3.6.1 JS Loss Formulation For a given LiDAR ray $\vec{r}$, we sample this ray with samples $s_i = \vec{o} + t_i\vec{d}$. z^* represents the depth of each along the ray, t_i represents the distance of each point along the direction vector to the origin, and w_i represents the weight predicted by the MLP for that point. We define a truncated Gaussian distribution K. The truncation parameter of this Gaussian distribution is ϵ, i.e., $K = N(0, (\frac{\epsilon}{3})^2)$ as the training distribution. Therefore, the target weights are $w_i^* = K(t_i - z^*)$. This JS loss can be defined as:

$$L_{JS}(\theta) = \|w_i^* - w_i\|_1 + \left\| 1 - \sum_i w_i \right\|_1 \quad (7)$$

However, in practical SLAM applications, both the continuous optimization and the constantly updated scene will lead to different regions having different degrees of learning, at this time, it will be inappropriate to use a uniform margin ϵ to adapt to the learning of the scene.

Thus, in this method, we learn LONER (Isaacson et al., 2023) and use the dynamic margin JS loss as the computational term of the loss function. For a region of unknown geometry, we use a larger margin to help converge quickly, while a smaller margin for a better-learned region. This allows the system to retain and refine learned regions while being able to learn new regions. We use JS divergence to measure the target distribution and the sample distribution on each ray to know how well the map on that ray is learned. Define the target distribution as $T = N(z^*, \sigma^*)$, where $\sigma^* = \epsilon_{min}/3$ define the sample distribution as $S = N(\overline{\mu}_w, \overline{\sigma}_w)$, where $\overline{\mu}_w$ and $\overline{\sigma}_w$ are signed as mean and standard deviation of the predicted weight along the ray respectively. The dynamic margin is defined as :

$$\epsilon_{dyn} = \epsilon_{min}(1 + \alpha J^*) \quad (8)$$

For J^* , its value depends on the JS score of T and S. If it is less than the minimum threshold JS_{min}, then J^* is 0. If it is greater than the maximum threshold JS_{max}, then J^* takes the maximum threshold, and the obtained score in other cases.

3.6.2 Depth Loss We follow the paradigm in (Rematas et al., 2022, Carlson et al., 2023) by adding a depth term to the loss function. Since LiDAR measures the exact depth, it is natural to include a depth loss. The depth loss is the difference between the rendered depth image and the depth image measured by LiDAR, i.e.

$$L_{depth}(\Theta) = ||\hat{D}(\hat{r}) - z^*||_2^2 \quad (9)$$

3.7 Meshing

To visualize the scene in an explicit way, a virtual LiDAR is placed at optimized Keyframe poses. When multiple weights fall on the same uniform mesh, the maximum value is retained. The marching cubes algorithm (Lorensen and Cline, 1998) is then used to form the mesh.

4. Experiments and Analysis

Figure 4. Example diagram. The example shows a typical underground area in a natural lava cave with dim lighting and complex pathways.

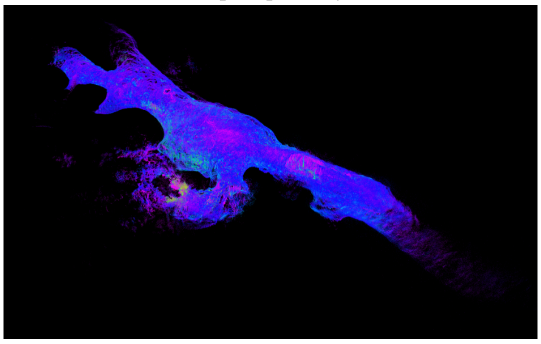

Figure 5. Schematic representation of experimental data. The collected experimental data were processed by the FASTLIO2 algorithm and visualized in Rviz.

In order to verify the effectiveness of our method, we conducted experiments in a lava cave data for testing, which was measured in a typical underground scene, consisting of an underground tunnel and some bifurcated paths, in line with our original intention of reconstructing the mesh of the underground scene as shown in Fig.4. In this experiment, we first started the helmet and miniature laptop to test the data link, and after the connection was stabilized, we carried out the data acquisition work. As shown in Fig.5, the irregular morphology and twisted shapes indicates the highly complex structure of the cave.

We implemented the proposed method on a desktop with Intel® Core™ i7-14700KF CPU and NVIDIA GeForce RTX 4090 GPU and achieves real-time efficiency. We used the proposed method to process the acquired data and perform mesh rendering and the rendered results are shown in Fig.6 and Fig.7. The figures indicate that our proposed method works well in this case.

Fig.6 shows our results for different cases of the same ray range with the mesh resolution. We can see that different ray range

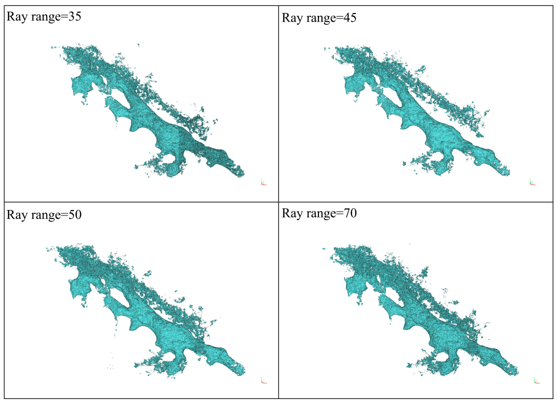

Figure 6. Comparison of different ray ranges with the same resolution. The values of ray range are 35, 45, 50, 70. The ray ranges used in the training and rendering are the same. These meshes are contained within the same bounding box size.

have a partial effect on the reconstruction of the scene. When the ray range is in the range of 35-70, the reconstructed mesh noise is the least in the ray range of 35, but there is a partial layering phenomenon, while with the increase of the ray range, the reconstructed mesh noise will increase a little, but the mesh details appear better. The ray range should be dynamically adjusted according to the actual situation of the map scene, when the scene is larger, a larger ray range is used, and vice versa.

Moreover, in Fig.7, we used multiple resolution rendering operations of 0.2, 0.3, and 0.4. In this figure, there’s a noticeable decrease in the detail and complexity among the meshes. The image with resolution = 0.2 results in a smoother and more detailed geometry, which is visually represented by continuity and a high level of surface detail. When you want to render a mesh in real time, you need to set a larger resolution and keyframe step to improve the rendering efficiency, when the computation time is enough, a smaller resolution and smaller keyframe step can render better results.

We also tested our method in WHU-Helmet dataset (Li et al., 2023). In this dataset, we chose the underground tunnel scenario as the test. In this test, we set ray range as 45 and the rendering resolution as 0.3, both of which we considered to be the most appropriate parameters in this scene. The reconstruction results are shown in Fig.8. In this scene, the mesh of the underground tunnel is supposed to be relatively smooth, and our method is also able to fit the smooth tunnel wall well. The right part of the Fig.8 shows the absolute error between the result and the ground truth value after reconstruction by our proposed method. In MeshLab, we use Hausdorff Distance to compare the tunnel reconstructed by the proposed method with the ground truth value and visualize it. In the visualization, we set the minimum value to 0, identified by blue color, and the maximum value to 1, identified by red color. The heat map shows that the deviation of our proposed reconstruction method compared to the ground truth value is within tolerance and good results are achieved.

Although the proposed method achieves satisfactory results in complex underground environments, there is still room for further improvement. First of all, there are some noise points on the top of the mesh, which is caused by the poor control of NeRF’s ray range parameter during rendering. Lowering the ray range can make the mesh less noisy, but at the same time, the phenomenon of layering will occur. Therefore, how to choose the appropriate ray range for trade-off is a question worth exploring. Secondly, NeRF is slow and not smooth enough when rendering the mesh in the back-end. Our future work will explore how to construct an incremental mesh with online speed and minimize the amount of data while guaranteeing geometric accuracy. Finally, using the real-time rendered mesh to promote front-end odometry thus forming a complete SLAM process including loop closure optimization, is also worth investigating.

5. Conclusion

In this paper, we introduce a novel approach for mesh reconstruction within a NeRF-based SLAM algorithm and evaluate its performance in an underground environment. To achieve an optimal balance between local positional accuracy and global optimization efficiency, we propose an innovative scan-block methodology designed to effectively retain local information. This approach aggregates multi-frame point clouds within a temporal window, with synchronized poses and point clouds aligned to the first frame. The aggregated scan-block is then processed as a cohesive unit in the back-end. Furthermore, to refine the reconstruction process, we incorporate NeRF training for precise mesh reconstruction of subterranean scenes. Leveraging the Marching Cubes algorithm, we efficiently extract a mesh directly from the implicit representation offered by NeRF, facilitating a more detailed and accurate portrayal of the underground environment. Experimental results validate that the proposed method strikes a compelling balance between reconstruction accuracy and computational efficiency. Additionally, the predictive interpolation capabilities of implicit neural fields provide further support for the reconstruction process.

6. Acknowledgement

This study was jointly supported by the National Natural Science Foundation Project (No. 42201477, No. 42130105).

References

Carlson, A., Ramanagopal, M. S., Tseng, N., Johnson-Roberson, M., Vasudevan, R., Skinner, K. A., 2023. Cloner: Camera-lidar fusion for occupancy grid-aided neural representations. *IEEE Robotics and Automation Letters*, 8(5), 2812–2819.

Deng, J., Wu, Q., Chen, X., Xia, S., Sun, Z., Liu, G., Yu, W., Pei, L., 2023. Nerf-loam: Neural implicit representation for large-scale incremental lidar odometry and mapping. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 8218–8227.

Franzius, J., Potts, D., 2005. Influence of mesh geometry on three-dimensional finite-element analysis of tunnel excavation. *International Journal of Geomechanics*, 5(3), 256–266.

Hashimoto, T., Saito, M., 2019. Normal estimation for accurate 3d mesh reconstruction with point cloud model incorporating spatial structure. *CVPR workshops*, 1.

Isaacson, S., Kung, P.-C., Ramanagopal, M., Vasudevan, R., Skinner, K. A., 2023. Loner: Lidar only neural representations for real-time slam. *IEEE Robotics and Automation Letters*.

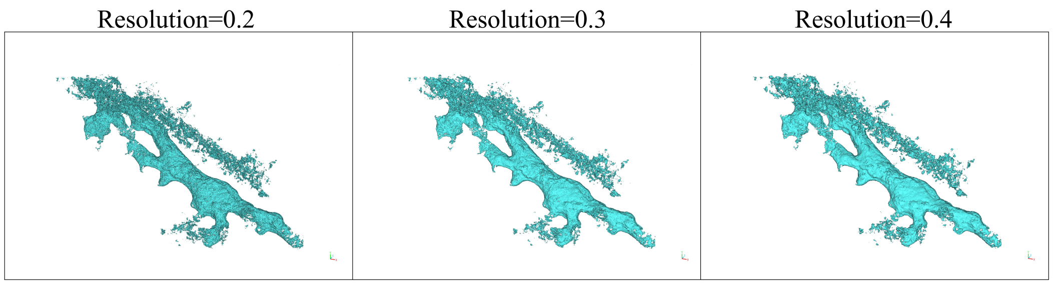

Figure 7. Comparison of different resolutions in the same ray range 45. The resolution sizes are 0.2, 0.3, 0.4, respectively. A fixed-size bounding box encloses the meshes.

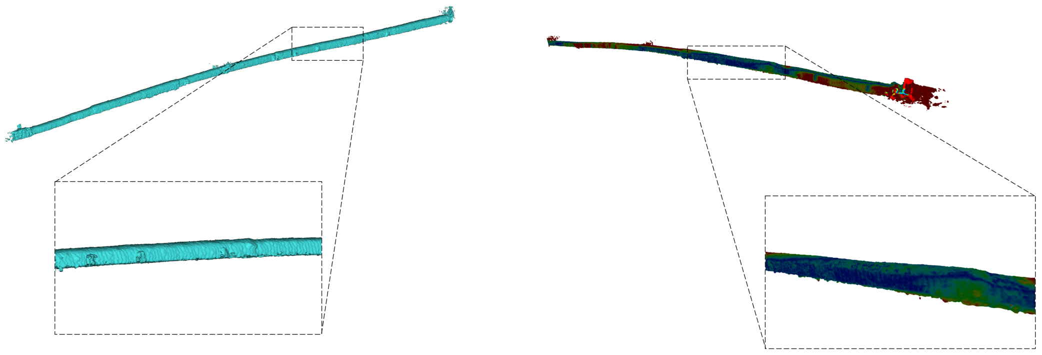

Figure 8. Underground tunnel reconstruction results. In order to highlight the focus of our work, we have only taken the part of underground tunnel for visualization.

Li, J., Wu, W., Yang, B., Zou, X., Yang, Y., Zhao, X., Dong, Z., 2023. WHU-Helmet: A helmet-based multi-sensor SLAM dataset for the evaluation of real-time 3D mapping in large-scale GNSS-denied environments. *IEEE Transactions on Geoscience and Remote Sensing*.

Li, J., Yang, B., Chen, Y., Wu, W., Yang, Y., Zhao, X., Chen, R., 2022. Evaluation of a compact helmet-based laser scanning system for aboveground and underground 3D mapping. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 43, 215–220.

Li, J., Yuan, S., Cao, M., Nguyen, T.-M., Cao, K., Xie, L., 2024. HCTO: Optimality-aware LiDAR inertial odometry with hybrid continuous time optimization for compact wearable mapping system. *ISPRS Journal of Photogrammetry and Remote Sensing*, 211, 228–243.

Long, N. Q., Buczek, M. M., Hien, L. P., Szlapińska, S. A., Nam, B. X., Nghia, N. V., Cuong, C. X., 2018. Accuracy assessment of mine walls’ surface models derived from terrestrial laser scanning. *International Journal of Coal Science & Technology*, 5, 328–338.

Lorensen, W. E., Cline, H. E., 1998. Marching cubes: A high resolution 3d surface construction algorithm. *Seminal graphics: pioneering efforts that shaped the field*, 347–353.

Park, S., Schöps, T., Pollefeys, M., 2017. Illumination change robustness in direct visual slam. *2017 IEEE international conference on robotics and automation (ICRA)*, IEEE, 4523–4530.

Rematas, K., Liu, A., Srinivasan, P. P., Barron, J. T., Tagliasacchi, A., Funkhouser, T., Ferrari, V., 2022. Urban radiance fields. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12932–12942.

Remondino, F., 2003. From point cloud to surface: the modeling and visualization problem. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34.

Ruan, J., Li, B., Wang, Y., Sun, Y., 2023. Slamesh: Real-time lidar simultaneous localization and meshing. *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 3546–3552.

Sucar, E., Liu, S., Ortiz, J., Davison, A. J., 2021. imap: Implicit mapping and positioning in real-time. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 6229–6238.

Xu, W., Cai, Y., He, D., Lin, J., Zhang, F., 2022. Fast-lio2: Fast direct lidar-inertial odometry. *IEEE Transactions on Robotics*, 38(4), 2053–2073.

Xu, W., Zhang, F., 2021. Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter. *IEEE Robotics and Automation Letters*, 6(2), 3317–3324.

Yang, B., Haala, N., Dong, Z., 2023. Progress and perspectives of point cloud intelligence. *Geo-spatial Information Science*, 26(2), 189–205.

Yang, B., Li, J., 2022. A hierarchical approach for refining point cloud quality of a low cost UAV LiDAR system in the urban environment. *ISPRS Journal of Photogrammetry and Remote Sensing*, 183, 403–421.

Zhu, Z., Peng, S., Larsson, V., Xu, W., Bao, H., Cui, Z., Oswald, M. R., Pollefeys, M., 2022. Nice-slam: Neural implicit scalable encoding for slam. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12786–12796.