Smart Structures and Systems, *Vol. 36, No. 5 (2025) 265-277*
https://doi.org/10.12989/sss.2025.36.5.265

Smartphone assisted 3D structural reconstruction and real-time scene segmentation modeling research

Yanzhi Qi [1,2,3a], Zhi Ding[*1,3] and Yaozhi Luo [2b]

[1] *Department of Civil Engineering, Hangzhou City University, Hangzhou, 310015, China*
[2] *Institute of Structural Engineering, Zhejiang University, Hangzhou, 310058, China*
[3] *Key Laboratory of Safe Construction and Intelligent Maintenance for Urban Shield Tunnels of Zhejiang Province, Hangzhou, 310015, China*

(*Received May 28, 2025, Revised November 15, 2025, Accepted December 9, 2025*)

Abstract. Since it is difficult to extract and understand the multi-dimensional spatial feature information from two-dimensional images, three-dimensional modeling and scene segmentation of structures are necessary. At present, most three-dimensional reconstruction technologies have the issues of expensive data acquisition equipment, long modelling time, and poor robustness under low texture. Considering the above concerns, this study proposed an integrated laser and visible light method based on the built-in lidar sensor of a smartphone to reduce the cost and achieve fast reconstruction of structural 3D point cloud models. In addition, a lightweight segmentation backbone network based on spatio-temporal redundancy downsampling mechanism was developed and deployed, which can realize real-time segmentation and information understanding of the environment. An unfurnished apartment was selected to verify the performance of the proposed method, and a floor plan was finally generated for comparison with the real values. Results show that the robustness of point cloud reconstruction was enhanced and the mIoU of scene segmentation on the test set reached 65.2% at 200FPS. The dimension of the generated floor plan has also obtained high accuracy (over 93% and mostly up to 97%), which is efficient for engineers to quickly and accurately understand on-site spatial structure information.

Keywords: 3D structural reconstruction; Lidar sensor; LSSNet; point cloud; scene segmentation

1. Introduction

Nowadays, the development of civil engineering is leading to more demands for machine and computer vision-based inspections. Quick acquisition of multi-dimensional spatial characteristic information is essential for on-site inspections of building facilities and efficient monitoring of dynamic changes (Xu and Stilla 2021). Traditional two-dimensional images include limited scene information, which scarcely conveys effective quantitative data for actual engineering projects. To achieve more detailed and intuitive structural spatial messages, 3D reconstruction and scene segmentation have become indispensable procedures (Hu *et al*. 2023, Liu *et al*. 2025).

Many methods have been developed to obtain 3D information of structural space models, among which stereo matching and laser scanning are considered to be the most versatile ways for depicting structural scenes (Braun and Borrmann 2019). Stereo matching mainly includes monocular depth estimation and Multiple View Stereo (MVS) technologies. Traditional machine learning-based monocular depth estimation applies Conditional Random Field (CRF) to directly estimate the depth value corresponding to each pixel of the image (Dijk and Croon 2019). Although this method is simple in algorithm structure, it often requires large amounts of depth-labelled data, which may result in higher acquisition cost and has the problem of scale uncertainty by using monocular images only. To get a higher level of details and better texture, Langguth *et al*. (2016) utilized the plane assumption to improve the reconstruction performance of weakly textured regions. Heo *et al*. (2022) presented a deep learning approach to automatically generate high-quality 3D point cloud models of civil infrastructure through implementing cropping, semantic segmentation, and implementing region extraction algorithms on large-format aerial images of cut slopes and dams captured by UAVs. However, the configuration and calibration of MVS methods are complex, its data processing consumes a lot of calculations and often requires GPU device acceleration, and the estimated depth accuracy was influenced by the image alignment.

The technique of Structure from Motion (SfM) (Ullman 1979) estimates 3D information from images of the time series by matching the disparity between the points. Although SfM can reconstruct structures from image sequences, its robustness degrades under low-texture or dynamic lighting conditions (Jin *et al*. 2003, Westoby *et al*. 2012). As an improved real-time approach to SfM, Simultaneous Localization and Mapping (SLAM) added sensing techniques (e.g., laser sensors or depth cameras) to reconstruct the whole scene (Debeunne and Vivet 2020, Macario Barros *et al*. 2022). However, the dependence on

*Corresponding author, Ph.D., Professor,
E-mail: dingz@hzcu.edu.cn
[a] Ph.D., E-mail: qiyanzhi@hzcu.edu.cn
[b] Professor, E-mail: luoyz@zju.edu.cn

ISSN: 1738-1584 (Print), 1738-1991 (Online)

high overlap between consecutive images and sensitivity to environmental texture still limit its performance. To enhance the performance of SLAM systems in low texture scenes, Zhu *et al*. (2024) proposed an RGB-D VSLAM algorithm that integrates a deep learning feature extraction network to reduce the average absolute trajectory error by 59%, and Liu *et al*. (2025) optimized the AGAST-based feature extraction algorithm to adaptively adjust the extraction threshold according to the gradient size of different data features, achieving an average AP value of 92.03%. While such methods effectively improve localization accuracy, they generally involve heavy network architectures and high computational costs, making them unsuitable for lightweight or mobile applications.

In order to eliminate false matching and improve model consistency, laser scanning has been proposed for 3D reconstruction application. Zhen *et al*. (2020) presented a LiDAR-enhanced SfM approach that combines point clouds with visual features in a joint optimization framework to reduce motion drift. Kim *et al*. (2023) used 3D scanning-based reverse engineering models to obtain defect data with high objectivity and reliability to improve the efficiency of the building diagnosis process. Although LiDAR technology has been proved to be effective in characterizing 3D scenes with high precision (Chiang *et al*. 2017, Tachella *et al*. 2019), the scanning equipment is extremely expensive (Li *et al*. 2020). To enable its use in low-cost applications, Gao and Peh (2016) proposed a smartphone-based planar laser distance sensor suitable for outdoor use with an accuracy of 6 cm at a distance of 5 m and a scan rate of 30 Hz. Jeftha and Shoko (2024) developed a low-cost alternative for multidisciplinary data collection based on laser scanning. Nevertheless, these solutions often sacrifice accuracy and scanning density, making them insufficient for precise reconstruction of fine indoor details.

The 3D point cloud data obtained by laser scanning directly reflects the geometry of the object surface, thus greatly simplifying indoor reconstruction and has been used in various tasks in the construction industry. During the building inspection process, the point cloud can assign 3D coordinate information to the target measurement points, but does not contain any semantic and topological information (Malinverni *et al*. 2019). To achieve a higher-level representation, RGB data and deep learning-based segmentation have been introduced (Weiss and Baret 2017, Yan *et al*. 2020, Lin *et al*. 2021, Chen *et al*. 2024). Su *et al*. (2015) proposed MVCNN to decompose 3D models into 2D multi-view images and then utilized CNN to realize segmentation. In order to achieve effective feature learning directly on point clouds, Maturana and Scherer (2015) put forward VoxNet to convert point clouds to voxel and then used 3D convolution kernel for convolution operation. Qi *et al*. (2017) proposed PointNet++, which extracted global features of point clouds using multilayer perception (MLP), and added self-adaptive density feature extraction to achieve local features. Although these techniques achieved significant progress, their large backbone networks and high computational demands lead to low segmentation efficiency, which restricts real-time applications on mobile or embedded devices (Geetha *et al*. 2024).

Existing 3D reconstruction and segmentation approaches still suffer from several challenges in practical civil engineering scenarios, including high computational costs, poor robustness in low-texture environments, expensive equipment, and lack of lightweight deployable solutions. To address these issues, this paper firstly proposed a method that fused laser and visible light to obtain point cloud data with more detailed textures. The method leverages the Metal programming interface for efficient real-time reconstruction, and develops a Lightweight Scene Segmentation Network (LSSNet) based on a spatio-temporal redundancy downsampling mechanism to achieve real-time semantic segmentation directly on mobile devices. Unlike conventional visual-LiDAR SLAM frameworks, the proposed Camera-LiDAR fusion approach focuses on efficient, lightweight, and frame-wise point cloud reconstruction without performing global optimization. It estimates the device motion through time-synchronized transformations and motion compensation rather than iterative SLAM-based optimization, thereby avoiding complex back-end processes such as bundle adjustment or loop closure. Through this lightweight reconstruction approach, rapid scene modeling and visualization can be achieved on mobile devices, and building floor plans can be automatically generated from the reconstructed and segmented indoor models, enabling engineers to quickly and accurately understand on-site structural information.

2. Laser and visible light integrated 3D reconstruction

This study proposes a rapid indoor 3D reconstruction and segmentation framework comprising three main stages. Firstly, a smartphone-based fusion framework integrates the built-in LiDAR and RGB camera is used to simultaneously capture depth and texture information. The Unscented Transform (UT) is employed for motion compensation to ensure spatial alignment between frames. Secondly, a Lightweight Scene Segmentation Network (LSSNet) is developed to perform real-time semantic extraction from the reconstructed point cloud on the mobile platform. Finally, the orthographic projection rectangle algorithm is applied to automatically convert the reconstructed 3D indoor scene into a 2D architectural floor plan, providing dimensional information for structural analysis.

2.1 Notation and frames

The proposed system involves multiple coordinate frames to describe the spatial relationships among the LiDAR sensor, RGB camera, smartphone body, and reconstructed 3D scene. To improve clarity and mathematical rigor, this subsection defines the coordinate frames, transformation rules, and the notational conventions adopted in this study. The definitions of superscripts and subscripts are explicitly stated to ensure consistency throughout the paper.

Table 1 Coordinate frame definitions

Symbol	Frame description	Axis definition/origin
ld	LiDAR sensor frame	Origin at LiDAR center; *x*-axis points forward, *y*-axis to the right, *z*-axis upward
cld	Camera-LiDAR fusion reference frame	Origin at camera optical center; aligned with LiDAR after calibration
sp	Smartphone base frame	Origin at IMU center; defined by device's internal coordinate system
2d	Image pixel coordinate frame	Origin at top-left of image; *x*-axis (*u*) rightward, *y*-axis (*v*) downward
ref	Global reference or world frame	Fixed inertial reference during motion compensation and mapping

Table 2 Notation conventions

Symbol	Meaning
$P_{c,i}$	The *i*-th LiDAR point in the *c*-th scan
T_{sp}^{ld}	Transformation from smartphone frame to LiDAR frame
R_{sp}^{ld}, t_{sp}^{ld}	Rotation and translation components of T_{sp}^{ld}
T_{ld}^{cld}	Transformation from LiDAR frame to Camera-LiDAR frame
T_{cld}^{2d}	Projection from 3D fusion frame to 2D image space
$Z_{i,j}^{ld}$	Position of the *j*-th point in the *i*-th LiDAR scan (in F^{ld})
$Z_{i,j}^{2d}$	Corresponding image coordinates of $Z_{i,j}^{ld}$
TS_{ref}	Reference timestamp for Camera-LiDAR synchronization
μ, Σ	Mean vector and covariance matrix in Unscented Transform
α, β, κ	Unscented Transform scaling parameters
$\psi(t, t+1)$	Structural Similarity Index (SSIM) between consecutive frames

Table 3 Coordinate frame definitions

Superscript and subscript definitions	Definition	Example
Superscript (^)	Denotes the reference frame in which a vector or transformation is expressed	Z^{ld}: point expressed in LiDAR frame
Subscript (_) on *T*	Denotes the source frame from which the transformation originates	T_{sp}^{ld}: from smartphone to LiDAR
Subscript (_) on *z* or *p*	Denotes index or timestamp	$P_{c,i}$: *i*-th point in scan *c*
T_b^a	Follows robotics convention: transform from frame *b* to frame *a*	$Z^a = T_b^a Z^b$
Time dependence	Indicated by (*t*) when needed	$Z^{ld}(t_i)$: point at timestamp t_i

2.2 Coordinate system transformation and motion compensation

Laser scanning acquires the distance to the object through the time lapse between the emitted and reflected pulse signals and the Time of Flight (ToF) method (Foix *et al*. 2011), thus presenting precise spatial positions of the target. Although laser sensors do not depend on surrounding light sources to capture objects in the physical scene, the robustness is still lower than that of stereo or structured light technologies. In addition, laser scanning can reflect the shape, attitude and position of real-world targets, but lack textural information (Van der Jeught and Dirckx 2016). Visible light cameras are characterized by high resolution and have access to RGB information from the environment, but they cannot directly obtain the depth of the target and perform poorly in low-light environments. In view of the above, this study proposed an indoor 3D modeling method that fused laser and visible light to overcome the drawbacks of each individual sensor.

The integrated method utilized the built-in LiDAR sensor of iPhone 12 Pro to capture 3D point clouds within a time stamp synchronized to the camera image. In order to accurately combine the information from the LiDAR sensor and smartphone cameras, the transformation between the real world and pixel coordinate systems was calibrated, and the motion compensation and occlusion handling were also performed. Due to the translation and rotation of the person holding the smartphone while scanning around the room, the end part of the point cloud was registered to a different position compared to the beginning part, as shown in Fig. 1. During the scan, the coordinate system of the LiDAR sensor

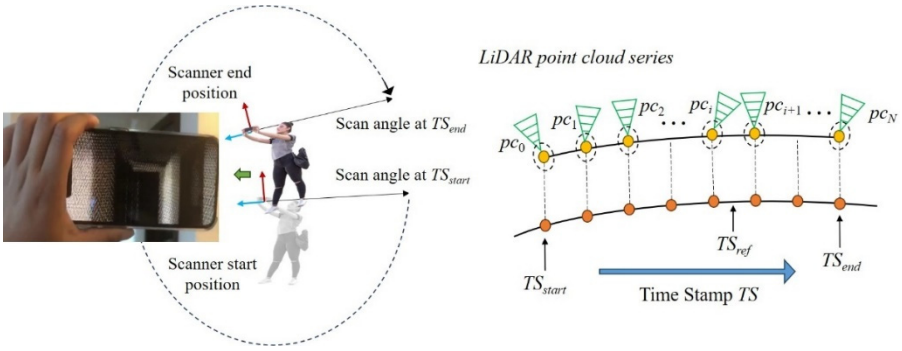


Fig. 1 The process of LiDAR point cloud motion compensation

moves with the person, but the camera images are instantaneous snapshots, so their timestamps do not coincide. Therefore, LiDAR point cloud series were first motion compensated and converted to the same reference frame with the camera images. Each set of LiDAR point clouds was transformed based on the estimated delta translations and rotations of the smartphone platform between the camera image reference time stamps TS_{ref} and the point cloud time stamps.

To establish a unified theoretical framework, the motion compensation was formulated as a probabilistic state estimation problem under a vision-LiDAR joint observation model. Specifically, the smartphone's 6-DoF motion was represented by a state vector $[x,\ y,\ z,\ \phi,\ \theta,\ \psi]$ and the temporal offset between LiDAR and camera data introduced uncertainty in both position and orientation. The proposed approach performs compensation by propagating this uncertainty through a nonlinear transformation using the Unscented Transform (UT) (Wan *et al*. 1999), thereby aligning all LiDAR points into the camera reference frame while maintaining statistical consistency.

UT is a deterministic sampling technique designed to estimate the statistics of a random variable that undergoes a nonlinear transformation. Unlike the Extended Kalman Filter (EKF), which linearizes the nonlinear system using Jacobian matrices and may introduce approximation errors, the UT directly propagates a small set of carefully chosen sample points (called sigma points) through the nonlinear function, achieving higher-order accuracy for the resulting mean and covariance estimation. Let the system state be represented by a random vector $x \in \mathbb{R}^n$ with mean $\bar{x}$ and covariance P_x, the UT generates $2n$+1 sigma points as follows

$$\begin{aligned} &\chi_0 = \bar{x}, \quad \chi_i = \bar{x} + (\sqrt{(n+\lambda)P_x})_i, \\ &\chi_{i+n} = \bar{x} - (\sqrt{(n+\lambda)P_x})_i \end{aligned} \tag{1}$$

where $\lambda = \alpha^2(n+\kappa) - n$, α and κ are scaling parameters.

The corresponding mean and covariance weights are computed as

$$\begin{aligned} &W_0^m = \frac{\lambda}{n+\lambda}, \quad W_0^c = \frac{\lambda}{n+\lambda} + (1-\alpha^2+\beta), \\ &W_i^m = W_i^c = \frac{1}{2(n+\lambda)} \quad (i = 1,\ldots,2n) \end{aligned} \tag{2}$$

where β is used to incorporate prior knowledge about the distribution. This study adopts $\alpha = 10^{-3}$, $\beta = 2$ (for Gaussian priors), and $\kappa = 0$.

Each sigma point is then propagated through the nonlinear motion model $y_i = f(\chi_i)$, The transformed mean $\bar{y}$ and covariance P_y are computed by weighted summation

$$\bar{y} = \sum_{i=0}^{2n} W_i^m y_i, \quad P_y = \sum_{i=0}^{2n} W_i^c (y_i - \bar{y})(y_i - \bar{y})^T \tag{3}$$

where W_i^m and W_i^c are the corresponding mean and covariance weights. The process noise was modeled as zero-mean Gaussian noise reflecting the sensor's intrinsic drift.

In this study, the nonlinear function $f(\cdot)$ represents the rigid-body transformation from the LiDAR coordinate frame to the camera frame, expressed as $p_c = R_{LC}p_L + t_{LC}$, where R_{LC} and t_{LC} denote the rotation and translation between sensors. The UT allows the uncertainty in motion parameters to be directly propagated through this transformation, enabling accurate prediction of the compensated LiDAR points without requiring Jacobian derivatives. Assuming that a scan consists of N sets of LiDAR point clouds with timestamps denoted as follow

$$\{p_{c,i}, TS_i^{pc}\}_{i=0}^{N-1} \tag{4}$$

where pc_i contains K measuring points $\{[x^{ld}, y^{ld}, z^{ld}, 1]^T_{i,j}\}_{j=0}^{K-1}$.

The reference time stamp TS_{ref} was selected as the common frame after sensor fusion, so it was set to coincide with the closest image. To estimate the motion of the moving smartphone, a sequence of linear velocity vectors were constructed based on monotonically increasing point cloud time stamps, as well as angular velocity vectors

$$z_{0:N-1}^v = \{z_i^v\}_{i=0}^{N-1}, \quad z_{0:N-1}^w = \{z_i^w\}_{i=0}^{N-1} \tag{5}$$

Supposing that the change of the person's moving speed during the time difference was negligible, z_i^v and z_i^w could be approximated using those with the closest timestamps to TS_i^{pc}, respectively. Furthermore, z_i^v and z_i^w were assumed to include independent and identically distributed zero-mean Gaussian noise with covariance matrices expressed as $\sum v$

and $\sum w$, respectively. The chronological jitter in TS_l^{pc} was modelled as zero-mean Gaussian noise with standard deviation σ_t, and the smartphone centric state at TS_{ref} was expressed as the following Gaussian variable

$$\boldsymbol{X}_{ref}^{sp} \sim \boldsymbol{N}(\bar{\boldsymbol{X}}_{ref}^{sp}, \sum_{ref}^{sp}) \tag{6}$$

Motion compensation for each set of corresponding point clouds of 3D LiDAR measuring points utilized predicted smartphone centric poses $\{\boldsymbol{X}_i^{sp} \sim \boldsymbol{N}(\bar{\boldsymbol{X}}_i^{sp}, \sum_i^{sp})\}_{i=0}^{N-1}$ at TS^{pc}, and for $i = 0, 1, \ldots, N-1$, the state mean and covariance matrix were decomposed by UT into a group of points

$$\{\chi_{i,r}^{sp}, \omega_{i,r}^{m}, \omega_{i,r}^{c}\}_{r=0}^{2d} \leftarrow UTD(\bar{\boldsymbol{X}}_{ref}^{sp}, \sum_{ref}^{sp}) \tag{7}$$

For $j = 0, 1, \ldots, K-1$, and $r = 0, 1, \ldots, 2d$, the point after motion compensation was calculated as

$$Z_{i,j,r}^{cld} = (T_{sp}^{ld})^{-1} \cdot \psi_{i,r}^{sp} \cdot T_{sp}^{ld} \cdot z_{i,j}^{ld} \tag{8}$$

where $\{\psi_{i,r}^{sp}\}_{r=0}^{2d}$ is a 4×4 homogeneous transformation matrices from the rotation and translation in each $\chi_{i,r}^{sp}$. The rigid transformation T_{sp}^{ld} converted the LiDAR point to the smartphone base frame and then encapsulate the motion, and finally, the point was converted back to the LiDAR coordinate system.

2.3 Camera-LiDAR projection and occlusion handling

The motion-compensated point clouds were taken as input to perform the projection of the camera image coordinate system. Before projection, each 3D point was translated from the LiDAR frame to the camera frame, and the external calibration between the two sensors was represented as a transformation matrix T_{cam}^{ld}

$$z^{cam} = T_{cam}^{ld} z^{ld} \tag{9}$$

where $z^{cam} = [x^{cam}, y^{cam}, z^{cam}, 1]^T$ indicates the 3D point that converted to camera frames. The pixel coordinates (u, v) were transformed to the image frame corresponding to z^{cam} in the camera frame based on the camera intrinsic parameters, and the projection function was defined as

$$\begin{bmatrix} u \\ v \end{bmatrix} = f_{proj}(z^{cam}) \tag{10}$$

The translation of motion-compensated point clouds $Z_{i,j,r}^{cld}$ from LiDAR frame to camera frame using T_{cam}^{ld} and the projection to the image frame can be combined through the following equation

$$\kappa_{i,j,r}^{cam} = f_{proj}(T_{cam}^{ld} \cdot Z_{i,j,r}^{cld}) \tag{11}$$

For $i = 0, 1, \ldots, N-1$, and $j = 0, 1, \ldots, K-1$, the image pixel projected from the LiDAR point cloud within pc_i can be obtained from the mean and covariance matrices by

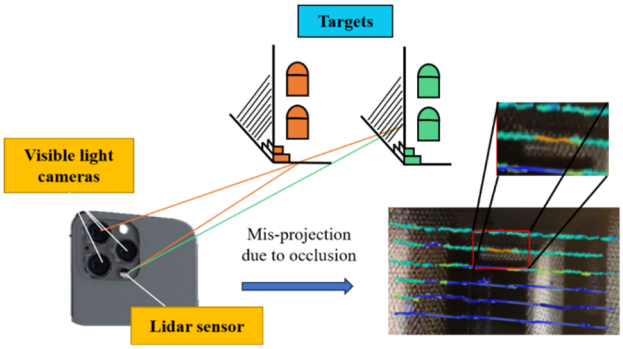


Fig. 2 Occlusion-induced point clouds mis-projection into images

$$\begin{bmatrix} \bar{u}_{i,j} \\ \bar{v}_{i,j} \end{bmatrix}, \quad \sum_{i,j}^{u,v} \leftarrow UTR(\{\kappa_{i,j,r}^{cam}, w_{i,r}^{m}, w_{i,r}^{c}\}) \tag{12}$$

In this study, the smartphone was equipped with three cameras and one built-in LiDAR sensor at different locations, which means that they are observing scenes from various viewpoints. The LiDAR sensor located below the camera can scan the targets that behind the obscuring structures, as shown in Fig. 2, however, the green target was occluded from the camera perspective in this case. Since this issue may lead to incorrect allocation and transmission of the information from the camera and the LiDAR sensor, a masking approach was applied to handle the occlusion when projecting a sparse point cloud into an image where the sensors are not collocated.

Firstly, the point clouds in the camera frame [x^{cam}, y^{cam}, z^{cam}] were sorted from near to far using the K-Nearest Neighbor (Taunk *et al*. 2019) before projecting to the image frame. Then a mask was created for each of those projected points to prevent other points being placed in the region. The mask is a rectangle with x_{mask} and y_{mask} dimensions depending on the vertical and horizontal resolution of LiDAR, and the gap between points can be calculated by the following equation

$$G = d_t \times tan(\theta_{s-h}) \tag{13}$$

where d_t refers to the distance to the target, θ_{s-h} denotes the vertical or horizontal angle between scan lines or consecutive points. Since the distance between the LiDAR and the camera is much smaller than that between the smartphone and the obscuring structure, x_G and y_G can be obtained according to camera intrinsic parameters and LiDAR angular resolution

$$x_G = f_x \, tan(\theta_h), \quad y_G = f_y \, tan(\theta_s) \tag{14}$$

For the cameras and LiDAR sensor of iPhone 12 pro used in this study, $\theta_s = 0.1°$ and $\theta_h = 1°$, so the dimensions of the mask are $y_G = 3$ pixels and $x_G = 21$ pixels. If a 3D point is placed in the mask region, it is considered occluded and not taken into account. Finally, the point colour of the occlused target projected onto the camera image was rectified after applying the masking approach.

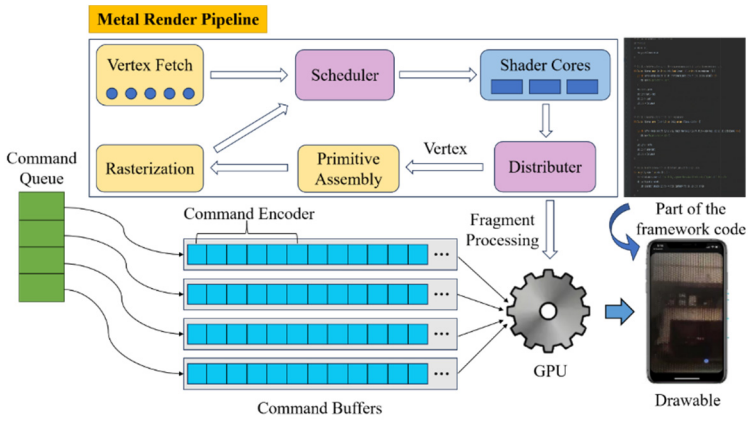


Fig. 3 Schematic of the 3D reconstruction framework

2.4 3D reconstruction framework based on Metal Interface

In this research, the built-in LiDAR scanner of iPhone 12 Pro combined with the A12Z Bionic processor and its neural engine were adopted to help sensing the depth of scenes. The LiDAR system emits a series of laser pulses in different parts of scenes in a short period of time, and the scanning range can be expanded to 5 meters. Nevertheless, depth data cannot provide the resolution required by some applications, neither can it perform scanning with higher accuracy like other photogrammetric techniques, so the 3D reconstruction framework was developed on the Application Programming Interface (API) for rendering and computation under the iOS platform. The APIs for mobile GPU rendering mainly include OpenGL, Vulkan, and Metal, etc. OpenGL has the longest history, but its design can no longer fully exploit the performance of the latest GPUs; Vulkan improves the control of programmers on GPUs, but has a relatively high learning cost and requires long development time. Compared to them, Metal has high development efficiency while maintaining well performance, which was applied to set up the reconstruction framework.

The program compilation used the Swift 3.0.2 open source language, Xcode 8.2.1 integrated development environment, and Metal Render Pipeline. From MTLDevice to MTLBuffer, the continuous camera image flow, tracking information, and other rendering programs have been combined. In order to execute the pipeline based on the specified data, command buffers were created to write the commands and commit them to the command queue. Once completed, the Metal API sent these commands to the GPU for computation, as displayed in Fig. 3, and the graphics pipeline on the GPU started as soon as it received all the commands and resources. After inputting the vertex buffer, each vertex element was first read sequentially in order, and then the texture of each vertex was computed by Vertex Shaders. Here the vertices were transformed between different coordinate systems to get the world position and depth of the specified camera point. The processed vertices were grouped and transferred to the element loading unit, and then performed rasterization and passed them back to the scheduler unit. A pixel slice shader function was created to receive the information by the vertex function such as lighting, texture coordinates, depth and color. Therefore, textures were added to the point clouds of the scene scanned by the LiDAR sensor. Through developing the 3D reconstruction framework based on the Metal Interface under the iOS platform, the mobile modeling has become more portable, automatic and efficient.

3. Lightweight scene segmentation backbone network

Following point cloud data acquisition, three main steps were considered to obtain the structural information of the reconstructed building, i.e., preprocessing, semantic segmentation, and performance evaluation. Semantic segmentation is to auto-segment the 3D point cloud to recognize and mark the various structural elements, such as floor, ceiling, wall, door, window, etc. After completing the model pre-training, each point will be classified into a predefined category based on common characteristics. Classical segmentation algorithms pursuit accuracy at the expense of efficiency, resulting in larger network parameters, higher storage space and hardware requirements, amd time-consuming model training (Li *et al.* 2023). In order to quickly achieve and understand the on-site structural information of scenes in buildings, this study developed a Lightweight Scene Segmentation Network (LSSNet) to mitigate time-consuming issues by eliminating structural redundancy, which significantly reduces the network parameters while maintaining the segmentation performance.

3.1 Spatio-temporal redundancy downsampling and feature extraction

The hierarchical neutral network was adopted to carry out global feature extraction in this research, which added a multi-level algorithm structure. First some points in the metric space were selected as centroids and then a region was formed around each centroid as an input sample. After obtaining a set of features, the region was expanded and the features extracted in the previous step were fed into the network as input. This process was to continuously extract

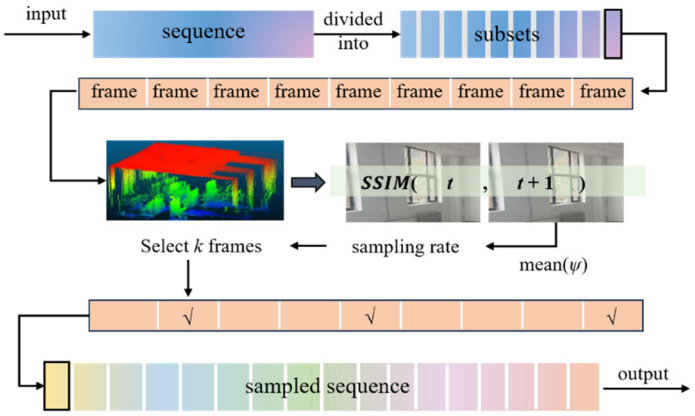


Fig. 4 Point cloud spatio-temporal redundancy frame downsampling

local features, then expanded the local scope, and finally got a set of global features. Each cluster of an abstraction layer of the network consists of 2 main parts: downsampling layer and grouping layer. The LiDAR point cloud dataset often shows significant redundancy due to the spatio-temporal correlation of the sequence. In order to reduce the training dataset size and redundant computations, the spatio-temporal redundancy frame downsampling mechanism was employed to determine the spatio-temporal redundancy by analyzing the spatial overlap in the time-continuous LiDAR frame sequences.

The process of point cloud spatio-temporal redundancy frame downsampling is shown in Fig. 4. The m point cloud frames in each sequence of the input were uniformly divided into $[m / n]$ subsets (containing n frames). For each frame within the subset, the corresponding RGB camera image was found at times t and $t + 1$ in the dataset. Then the similarity $\psi(t, t+1)$ between temporally neighboring frames was computed by the Structural Similarity Index (SSIM) (Li and Bovik 2009) to detect temporal redundancy at moment t. SSIM perceives changes mainly through three characteristics of the image: luminance, contrast ratio and structure. Let x and y denote camera images at times t and $t+1$, respectively, and each image has N pixels, the luminance can be obtained by averaging the values of all pixels

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i, \qquad \mu_y = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{15}$$

The contrast ratio is calculated by pixel standard deviation

$$\sigma_x = \left(\frac{1}{N-1}\sum_{i=1}^{N}(x_i-\mu_x)^2\right)^{\frac{1}{2}}, \qquad \sigma_y = \left(\frac{1}{N-1}\sum_{i=1}^{N}(y_i-\mu_y)^2\right)^{\frac{1}{2}} \tag{16}$$

The structure compares the normalized x - μ_x / σ_x with y - μ_y / σ_y, which can be measured by the following correlation coefficient

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i-\mu_x)(y_i-\mu_y) \tag{17}$$

Combining the contrast functions for luminance, contrast and structure, SSIM can be obtained as follow

$$\psi(x,y) = \left(\frac{2\mu_x\mu_y + C_1}{{\mu_x}^2 + {\mu_y}^2 + C_1}\right)^{\alpha} \cdot \left(\frac{2\sigma_x\sigma_y + C_2}{{\sigma_x}^2 + {\sigma_y}^2 + C_2}\right)^{\beta} \cdot \left(\frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}\right)^{\gamma} \tag{18}$$

where α, β, and γ represent the proportion of different characteristics, respectively.

In this study, the uncertainty associated with the SSIM-based frame similarity and its influence on the redundancy estimation were analyzed to ensure robust downsampling. To account for variations caused by illumination fluctuation and motion blur, a Gaussian uncertainty model was introduced for $\psi(t, t + 1)$, expressed as $\tilde{\psi}(t, t + 1) = \psi(t, t + 1)+\varepsilon$, where $\varepsilon \sim N(0, \sigma^2\psi)$. The variance $\sigma^2\psi$ was empirically estimated through repeated scans under different environmental conditions. Based on this uncertainty, a confidence interval was used to determine whether two consecutive frames were redundant or unique. Furthermore, sensitivity analysis was conducted on the parameters n (subset size) and the SSIM threshold τ, revealing that reconstruction accuracy was most sensitive to τ when $\tau <$ 0.75, where relative mIoU variation exceeded 4%.

The presence of temporal redundancy was estimated by the average of the similarity scores between all neighboring frames in the current subset. The sampling rate was then determined based on the average similarity of the frame selection within that subset. In this way, the higher the spatiotemporal redundancy ($\psi \rightarrow 1$), the lower the sampling rate. It was repeated for all subsets in each point cloud sequence to construct the final set of subsampled LiDAR frames used for training. To make the model more robust when the points are sparse, areas of different scales were sent for feature extraction, and then those features were concatenated as the central point. The random input dropout (DP) method was to perform on the point set, and the ratio was set to 90%.

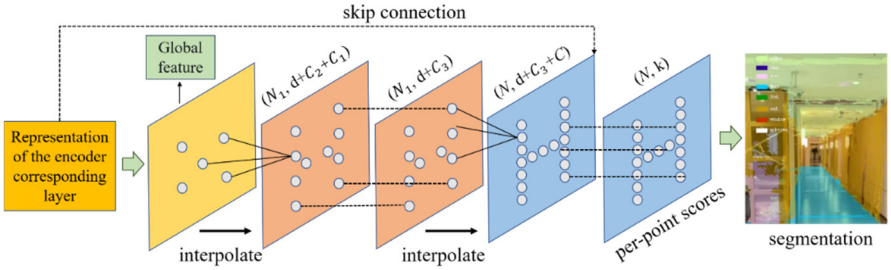


Fig. 5 Schematic diagram of backward interpolation and skip connection. (N: the number of output points, d: the coordinate dimension, C: the feature dimension, Ni: the number of input points, and Ci: the input feature dimension from upper layer)

3.2 Backward interpolation and skip connection

The depthwise separable convolution (Jang *et al*. 2023) was applied as decoder for segmentation, which significantly reduced trainable parameters and computational cost compared to conventional convolution operations. After spatio-temporal redundancy downsampling to get the global feature, the discriminative depthwise feature was then obtained by back interpolation and skip connection, as shown in Fig. 5. Assuming that the set of yellow rectangular points P_1 is $N_1 \times C$, the set of orange rectangular points P_2 is $N_2 \times C_2$. Since the decoder is an upsampling process, $N_2 > N_1$ and interpolated according to the following equation

$$f^{(j)}(x) = \frac{\sum_{i=1}^{k} \omega_i(x) f_i^{(j)}}{\sum_{i=1}^{k} \omega_i(x)}, \quad \omega_i(x) = \frac{1}{d(x, x_i)^P}, \quad j = 1, \ldots, C \tag{19}$$

For each point in P_2, k points in P_1 that are closest to it in the original point cloud coordinate space x_1, …, x_k were found. Then the features of x_1, …, x_k were weighted and summed to achieve the features of x. The weight is inversely correlated with the distance between x and x_1, …, x_k, namely, the farther away the point is, the less it contributes to the x-feature. The features obtained by backpropagation were from the upper layers of the decoder and were considered to be global information, which still lacked local information to get discriminative depthwise features. Hence, the skip connection was used to splice the representation of the previous encoder corresponding layers in order to achieve local information.

The loss function of LSSNet mainly contains two parts: the regular cross-entropy classification loss L_{cls}, and the feature transformation matrix loss L_{reg}. The dimension of the feature transformation matrix is large so regular terms are added to ensure the training stability of the network, the constrained generated feature matrix is orthogonal and the corresponding loss function is shown as follow

$$L_{reg} = ||I - AA^T||_F^2 \tag{20}$$

In the cross-entropy loss between foreground and background, $y \in \{\pm 1\}$ specifies the ground-truth class and $p \in [0, 1]$ is the model's estimated probability for the class with label $y = 1$. p_t is defined as

$$p_t = \begin{cases} p, & \text{if } y = 1 \\ 1 - p, & \text{otherwise} \end{cases} \tag{21}$$

A weighting factor is added to the cross entropy loss, when an example is misclassified and p_t is small, the modulating factor is near 1 and the loss is unaffected, and if $p_t \to 1$, the factor goes to 0 and the loss for well-classified examples will be down-weighted. Finally, the total loss function is defined as

$$L = -(1 - p_t)^{\gamma} log(p_t) + L_{reg} \tag{22}$$

4. Experimental procedures and result discussions

4.1 Field structural reconstruction and dataset creation

In order to verify the performance of the proposed method, an unfurnished apartment was selected and scanned using the integrated laser and visible light 3D reconstruction framework developed on an iPhone 12 pro smartphone. The apartment has a floor area of about 150 square metres and the entire indoor reconstruction took about 3 minutes, including the time of walking around the room for scanning and data exporting for visualisation. Fig. 6 displays the reconstructed 3D point cloud model in the Meshlab visualisation software from the top and front views, as well as some specific examples of the scanned rooms. As shown in the picture, the floor, ceiling and walls have been well reconstructed, but the point clouds at the joints and windows were partially missing. The reason could be that some overlapping areas were not well matched during the scanning process, and the reflective strengths of glass and other materials are different.

The speed of indoor 3D reconstruction of the proposed method was relatively fast mainly for three reasons as following. Firstly, the points were directly stored in the memory instead of performing I/O operations. Secondly, there was almost no large amount of computation in the function of collecting points at a single frame rate to directly obtain the 3D coordinates of points in space. Last but no least, the compute unit could batch processed multiple vertices at once, up to the number of shader cores it contained, until the current vertex processing stage was complete. Once the vertex processing was completed, the

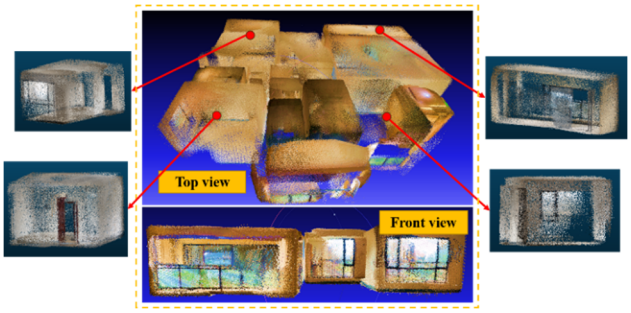


Fig. 6 The apartment reconstructed 3D point cloud visualisation model

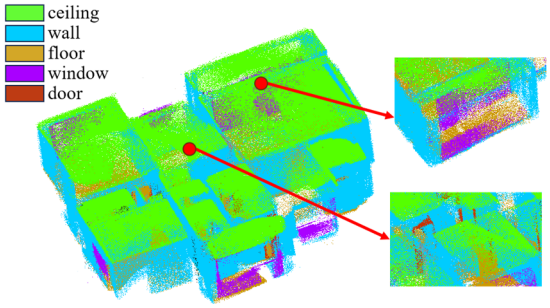


Fig. 7 Segmentation result of the reconstructed apartment point cloud model

buffer was emptied and prepared for the next batch of vertices processing.

The reconstructed apartment 3D model includes six channels (X, Y, Z, R, G, B) of spatial coordinates and colors with a total of 7179,646 points. As the point cloud scanned by LiDAR has the characteristics of disorder, asymmetry, and unevenness due to the influence of the target distance, it may lead to the unclear and unified data structures and increase the difficulty of subsequent segmentation (Woo *et al*. 2002). In order to set up effective dataset for training LSSNet model, the preprocessing and subsampling oprations were performed on the original point cloud data. Gaussian filtering was applied to remove the outliers and noise points to smooth the irregular density of data. Then Octree method was used to subsample the large number of points for reducing the computation and redundancy, as well as normalization. Finally, data augmentation such as rotation, flipping, and varing degrees of occlusion were performed to improve the robustness and generalisation. In addition, an open-source indoor 3D point cloud dataset with pixel-level semantic annotation developed by Stanford University has also been added for training (Armeni *et al*. 2016). All point cloud models were labelled with 5 semantic elements (wall, ceiling, floor, door, and window), and the number of training and testing building models were 4790 and 960, respectively.

4.2 Real-time scene segmentation and performance comparison

The LSSNet model training process was performed by running Pytorch 1.1.0, cuDNN 7.4.1, CUDA 10.0 on a computer equipped with Intel(R) Core (TM) i9-9900K@ 3.60 GHz CPU and 11 GB NVIDIA GeForce RTX 2080 Ti. At the beginning of training, the weight of each class was initialized. In this study, the minibatch size for network training was set to 16 and the training epoch was 50. To facilitate parameter updates between iterations, the raw momentum was set to 0.1, and the learning rate was 0.001. The training time using GPU mode took about 6 hours, and the visualised segmentation result of the apartment reconstructed model is shown in Fig. 7, where green, blue, yellow, purple, and red represent ceiling, wall, floor, window, and door, respectively. The Intersection over Union (IoU) and mean Intersection over Union (mIoU) are often used to evaluate the segmentation performance of algorithm models. Therefore, this study applied IoU of different classes and mIoU as norms to evaluate the precision. The formula of IoU and mIoU are respectively shown in Eqs. (23)-(24), where TP means true-positive sample, FP refers to false-positive sample, FN represents false-negative sample, and k refers to the number of element classes (k = 5 in this case).

$$\mathrm{IoU} = \frac{TP}{TP + FP + FN} \tag{23}$$

$$\mathrm{mIoU} = \frac{1}{k+1}\sum_{i=0}^{k} Io\,U_i \tag{24}$$

Table 4 shows the training and validation IoU of each category and mIoU of the whole environment. Among them, the values of ceiling and wall are higher than other classes, which can achieve over 65% IoU on validation set. The values of structural openings such as windows and

Table 4 The training and validation IoU of each class and mIoU of the whole scene

	Ceiling	Floor	Wall	Window	Door	Whole scene
Traing IoU/mIoU(%)	86.46	85.20	89.83	82.48	81.41	87.5
Validation IoU/mIoU(%)	65.38	63.27	68.72	64.21	62.88	66.8

Table 5 Metrics for point-based classification

Metrics	Ceiling	Floor	Wall	Window	Door
Average class acc.	63.65	63.29	52.39	58.86	62.35
Global class acc.	65.64	66.46	59.76	62.81	64.38

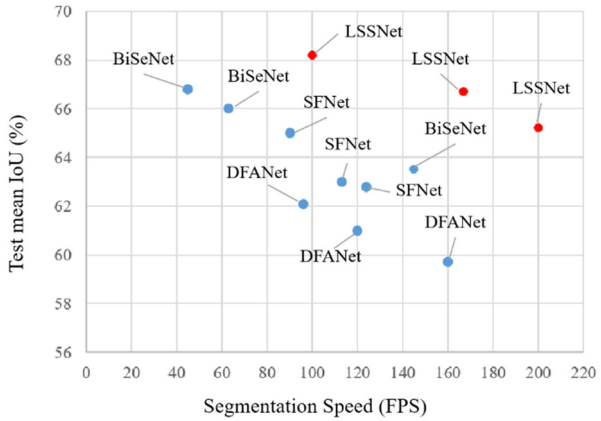


Fig. 8 Speed-Accuracy performance comparison on the test set. (the proposed method is presented in red dots while other methods are presented in blue dots)

doors are relatively lower, but can still reach more than 60%. The mIoU of the whole scene is 66.8%, which indicates that the developed segmentation network performs well on the dataset. The same indexes were also evaluated based on voxels, Table 5 shows the precision (point-based) of the five classes. Average accuracy (Acc.) refers to the average of the models' pixel classification accuracy in each category, and Global accuracy denotes the number of correctly predicted pixels divided by the total number of pixels. The results show that using the proposed reconstruction framework and LSSNet can achieve good segmentation of the environment, thus helping engineers to quickly understand on-site spatial structure information.

The segmentation speed is an important indicator of algorithm efficiency for lightweight networks. This study selected several common lightweight segmentation algorithms (BiSeNet (Yu *et al*. 2018), DFANet (Li *et al*. 2019), and SFNet (Lee *et al*. 2019)) to compare their performance on the test set with the proposed LSSNet. The speed-accuracy of different networks are displayed in Fig. 8. The proposed method achieves 65.2% mIoU on the indoor point cloud test set at 200FPS on 2080Ti, enabling real-time segmentation on site scenes. Meawhile, it achieves the highest mIoU of 68.2% at 97FPS, which improves 9.8% than that of DFANet at almost the same speed. At the same accuracy, the segmentation speed of LSSNet is three times faster than that of BiSeNet. Moreover, LSSNet outperforms SFNet in both accuracy and segmentation speed.

4.3 Floor plan generation and accuracy analysis

In order to obtain more straightforward and clearer structural information, a section of the middle height of the 3D point cloud model was intercepted horizontally, including the location of holes such as windows and doors, and then projected onto the same horizontal plane as the coordinate system to generate the building floor plan. In this experiment, the procedure of processing point cloud data and generating the floor plan were implemented by Python and Open3D functions. The floor plan generated by direct projection of the point cloud cross section was rough and has the problem of uneven line density, overlapping and pieces missing. Therefore, an orthographic projection rectangle method was proposed to optimize the building floor plan and facilitate the calculation of the length and width of each room.

The point cloud data (Fig. 9(a)) was firstly imported to get the (X, Y, Z) coordinates of all points (Fig. 9(b)), and then voxel grid filter was performed with the voxel size set as 0.5 to remove noise points. This could achieve downsampling without destroying the geometric structure of the point cloud, and the sparse point cloud after noise reduction is shown in (Fig. 9(c)). The general equation for the 3D space plane is presented as follows

$$A \cdot x + B \cdot y + C \cdot z = D \tag{25}$$

Assume that the coordinates of the point not in the plane are (x_0, y_0, z_0) and the coordinates of its projection point in the plane are (x_t, y_t, z_t). Since the projected point to the current point is perpendicular to the plane, according to the perpendicularity constraint, y_t and z_t satisfy the following conditions

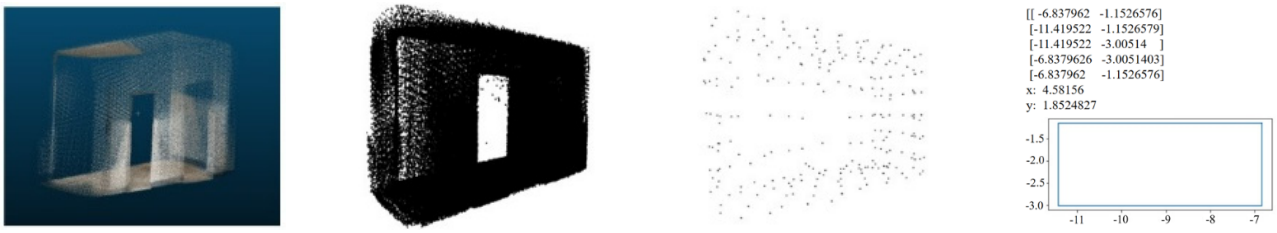
$$y_t = \left(\frac{B}{A}\right)(x_t - x_0) + y_0, \quad z_t = \left(\frac{C}{A}\right)(x_t - x_0) + z_0 \tag{26}$$

Substituting Eq. (26) into Eq. (25) yields

$$x_t = \frac{(B^2 + C^2)x_0 - A(By_0 + Cz_0 + D)}{A^2 + B^2 + C^2} \tag{27}$$

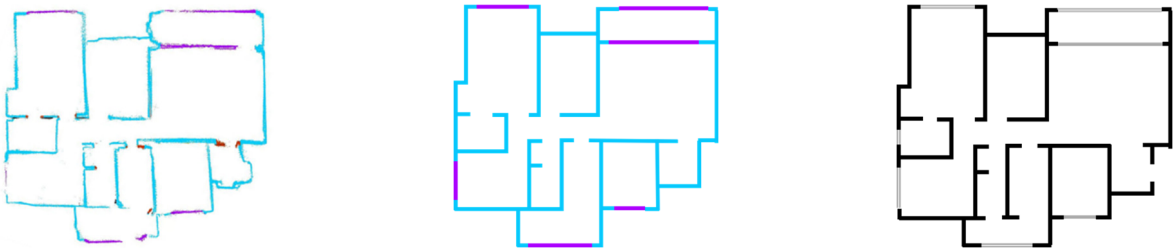
The substituting Eq. (27) into Eq. (26) yields

$$\begin{cases} y_t = \dfrac{(A^2 + C^2)y_0 - B(Ax_0 + Cz_0 + D)}{A^2 + B^2 + C^2} \\ z_t = \dfrac{(A^2 + B^2)z_0 - C(Ax_0 + By_0 + D)}{A^2 + B^2 + C^2} \end{cases} \tag{28}$$



(a) 3D Re-constructed room (b) Imported point cloud data (c) Sparse point cloud after Voxel Filtering (d) Projected result

Fig. 9 Orthographic projection rectangle method for floor plan optimisation



(a) Unprocessed direct-projection floor plan (b) Orthographic projection rectangle floor plan (c) Ground truth floor plan

Fig. 10 Comparison of the generated building floor plan with the ground truth

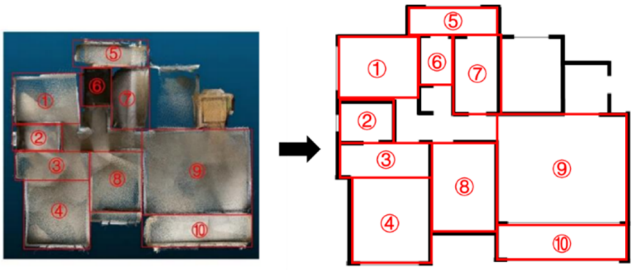


Fig. 11 Diagram of the area division

Finally, the coordinate system was rotated according to the outer-most point and the coordinate axis angle to get the orthographic projection rectangle of the apartment model's each part (Fig. 9(d)). After the voxel-filtered point cloud was projected onto a horizontal plane, each connected region corresponding to an individual room or corridor was detected using Euclidean clustering. The boundary points of each region were then extracted through a convex hull algorithm, and a minimum bounding rectangle was fitted to each cluster to obtain the orthographic projection rectangle for that part of the apartment. Linking these parts together can obtain the optimized orthographic projection rectangle floor plan (Fig. 10(b)), which is much closer to the ground truth than the direct projection (Fig. 10(c)). The comparison between these two results visually demonstrates the improvement of geometric regularity and completeness after applying the orthographic projection rectangle method. However, some of the locations of opening holes were concatenated rather than recognised and represented, which could be due to the classification errors in the point cloud.

To further quantify the reconstruction accuracy, the orthographic projection rectangle method was used to derive the geometric dimensions of each enclosed area. The outlines of the generated floor plan were analyzed by detecting the intersection points of the rectangular edges, and the corresponding coordinates were used to calculate the length and width of each space based on the distance between boundary points. The entire apartment model was then divided into ten areas according to the internal structural layout, as shown in Fig. 11. By measuring the dimensions of those areas and comparing them with the ground truth value, the accuracy of modeling can be quantified. Using the above measuring method, the length and width information of the ground truth L_{gt}, W_{gt} and the generated building floor plan L_{fp}, W_{fp} are shown in the table below, where the numbers ①~⑩ represent ten areas, respectively.

According to Table 6, it can be found that most of the distance errors can be controlled within 2%. The longer the distance, the greater the error that may occur. The lack of point cloud information in the data collection process of areas ⑥ and ⑦ may also result in large errors. Nevertheless, the accuracy of dimensions can still be achieved up to over 93%, mostly above 97%, which indicates that the indoor reconstruction and segmentation has obtained high accuracy through the proposed algorithms and techniques.

Table 6 The training and validation IoU of each class and mIoU of the whole scene

Area number	L_{gt} (mm)	L_{fp} (mm)	Accuracy (%)	W_{gt} (mm)	W_{fp} (mm)	Accuracy (%)
①	4200	4114	97.95	3200	3109	97.16
②	2800	2824	99.14	2200	2161	98.23
③	4500	4582	98.18	1800	1852	97.11
④	4400	4324	98.27	4200	4121	98.12
⑤	4200	4123	98.17	1500	1541	97.27
⑥	2400	2389	99.54	1800	1679	93.28
⑦	4000	3933	98.33	2400	2246	93.58
⑧	4200	4156	98.95	3400	3314	97.47
⑨	6900	6721	97.40	5600	5428	96.93
⑩	6900	6807	98.65	2000	2035	98.25

5. Conclusions

In this study, an integrated laser and visible light reconstruction method was proposed to obtain 3D point cloud model of indoor structures with more detailed colour textures. Unlike conventional SLAM methods that rely on global optimization and high computational cost, the proposed approach performs lightweight frame-level fusion using the built-in LiDAR and RGB camera of a smartphone, enabling fast, low-cost, and calibration-free 3D modeling. The novelty of this work can be summarized in three main aspects:

- A smartphone-based Camera-LiDAR integration strategy incorporating the Unscented Transform for accurate motion compensation and uncertainty propagation.
- The Lightweight Scene Segmentation Network (LSSNet) with spatio-temporal redundancy down-sampling is developed to achieve real-time segmentation on mobile devices.
- Through the automatic orthographic projection rectangle algorithm, 2D floor plans can be generated from the reconstructed 3D scenes, with dimensional errors controlled within 7%.

Experimental validation on an unfurnished apartment demonstrates that the proposed system achieves real-time segmentation at 200 FPS with an mIoU of 65.2%, while maintaining geometric accuracy in reconstructed models. This research provides a practical solution for low-cost structural modeling and on-site spatial analysis in civil engineering applications, but it still face some shortcomings, such as missing points at the joints and windows in the reconstructed model, and incorrect identification of some openings as walls, etc. In future work, a more accurate and detailed 3D reconstruction technology will be investigated, as well as the feature visualization and quantification on the structural components.

Acknowledgments

This work was supported by Key Research and Development Program of Zhejiang (Grant No. 2023C03182), and National Natural Science Foundation of China (Grant No. 52178400 and No. 52278418).

References

Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M. and Savarese, S. (2016), "3d semantic parsing of large-scale indoor spaces", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, June.

Braun, A. and Borrmann, A. (2019), "Combining inverse photogrammetry and BIM for automated labeling of construction site images for machine learning", *Autom. Constr.*, **106**, p. 102879. https://doi.org/10.1016/j.autcon.2019.102879

Chen, R., Wu, J., Luo, Y. and Xu, G. (2024), "PointMM: point cloud semantic segmentation CNN under multi-spatial feature encoding and multi-head attention pooling", *Remote Sens.*, **16**(7), p. 1246. https://doi.org/10.3390/rs16071246

Chiang, K.W., Tsai, G.J., Li, Y.H. and Sheimy, N.E. (2017), "Development of LiDAR-based UAV system for environment reconstruction", *IEEE Geosci. Remote Sens. Lett.*, **14**(10), 1790-1794. https://doi.org/10.1109/LGRS.2017.2736013

Debeunne, C. and Vivet, D. (2020), "A review of visual-LiDAR fusion based simultaneous localization and mapping", *Sensors*, **20**(7), p. 2068. https://doi.org/10.3390/s20072068

Dijk, T.V. and Croon, G.D. (2019), "How do neural networks see depth in single images?", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, June.

Foix, S., Alenya, G. and Torras, C. (2011), "Lock-in time-of-flight (ToF) cameras: A survey", *IEEE Sens. J.*, **11**(9), 1917-1926. https://doi.org/10.1109/JSEN.2010.2101060

Furukawa, Y. and Ponce, J. (2009), "Accurate, dense, and robust multiview stereopsis", *IEEE Trans. Pattern Anal. Mach. Intell.*, **32**(8), 1362-1376. https://doi.org/10.1109/TPAMI.2009.161

Gao, J.H. and Peh, L.S. (2016), "A smartphone-based laser distance sensor for outdoor environments", *Proceedings of the IEEE International Conference on Robotics and Automation*, Stockholm, Sweden, May.

Geetha, G.K., Lee, S., Lee, J. and Sim, S.H. (2024), "Long-term shape sensing of bridge girders using automated ROI extraction of LiDAR point clouds", *Smart Struct. Syst., Int. J.*, **33**(6), 399-414. https://doi.org/10.12989/sss.2024.33.6.399

Heo, J.H., Kwon, J.H., Khudoyarov, S. and Kim, N. (2022), "Deep learning approach to generate 3D civil infrastructure models using drone images", *Smart Struct. Syst., Int. J.*, **30**(5), 501-511. https://doi.org/10.12989/sss.2022.30.5.501

Hu, D., Gan, V.J. and Yin, C. (2023), "Robot-assisted mobile scanning for automated 3D reconstruction and point cloud semantic segmentation of building interiors", *Autom. Constr.*, **152**, p. 104949. https://doi.org/10.1016/j.autcon.2023.104949

Kim, S., Lee, J.M. and Kim, S. (2023), "Safety diagnosis process for deteriorated buildings using a 3D scan-based reverse engineering model", *Smart Struct. Syst., Int. J.*, **31**(1), 79-88. https://doi.org/10.12989/sss.2023.31.1.079

Jang, J.G., Quan, C., Lee, H.D. and Kang, U. (2023), "Falcon: lightweight and accurate convolution based on depthwise separable convolution", *Knowl. Info. Syst.*, **65**(5), 2225-2249. https://doi.org/10.1007/s10115-022-01818-x

Jeftha, K.J. and Shoko, M. (2024), "Mobile phone based laser scanning as a low-cost alternative for multidisciplinary data collection", *South Afr. J. Sci.*, **120**(11/12). https://doi.org/10.17159/sajs.2024/15437

Jin, H., Favaro, P. and Soatto, S. (2003), "A semi-direct approach to structure from motion", *Vis. Comput.*, **19**(6), 377-394. https://doi.org/10.1007/s00371-003-0202-6

Langguth, F., Sunkavalli, K., Hadap, S. and Goesele, M. (2016), "Shading-aware multi-view stereo", *Proceedings of the Computer Vision*, Amsterdam, The Netherlands, October.

Lee, J., Kim, D., Ponce, J. and Ham, B. (2019), "SFNet: Learning Object-Aware Semantic Correspondence", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, June.

Li, C. and Bovik, A.C. (2009), "Three-component weighted structural similarity index", *Image Qual. Syst. Perform. VI*, **7242**, 252-260. https://doi.org/10.1117/12.811821

Li, H., Xiong, P., Fan, H. and Sun, J. (2019), "DFANet: Deep Feature Aggregation for Real-Time Semantic Segmentation", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, June.

Li, D., Liu, J., Feng, L., Zhou, Y., Liu, P. and Chen, Y.F. (2020), "Terrestrial laser scanning assisted flatness quality assessment for two different types of concrete surfaces", *Measurement*, **154**,

p. 107436. https://doi.org/10.1016/j.measurement.2019.107436

Li, L., Shum, H.P. and Breckon, T.P. (2023), "Less is more: Reducing task and model complexity for 3d point cloud semantic segmentation", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vancouver, Canada, June.

Lin, C.C., Kuo, C.H. and Chiang, H.T. (2021), "CNN-based classification for point cloud object with bearing angle image", *IEEE Sens. J.*, **22**(1), 1003-1011. https://doi.org/10.1109/JSEN.2021.3130268

Liu, Y., Dong, S., Wang, S., Yin, Y., Yang, Y., Fan, Q. and Chen, B. (2025), "Slam3r: Real-time dense scene reconstruction from monocular rgb videos", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Nashville, TN, USA, June.

Liu, Y., Feng, Z., Zhang, H. and Dong, W. (2025), "Post-integration based point-line feature visual SLAM in low-texture environments", *Sci. Rep.*, **15**(1), p. 14606. https://doi.org/10.1038/s41598-025-97250-6

Macario Barros, A., Michel, M., Moline, Y., Corre, G. and Carrel, F. (2022), "A comprehensive survey of visual slam algorithms", *Robotics*, **11**(1), p. 24. https://doi.org/10.3390/robotics11010024

Malinverni, E.S., Pierdicca, R., Paolanti, M., Martini, M., Morbidoni, C., Matrone, F. and Lingua, A. (2019), "Deep learning for semantic segmentation of 3D point cloud", *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.*, **42**, 735-742. https://doi.org/10.5194/isprs-archives-XLII-2-W15-735-2019

Maturana, D. and Scherer, S. (2015), "Voxnet: A 3d convolutional neural network for real-time object recognition", *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, Hamburg, Germany, September.

Qi, C.R., Yi, L., Su, H. and Guibas, L.J. (2017), "Pointnet++: Deep hierarchical feature learning on point sets in a metric space", *Proceedings of the International Conference on Neural Information Processing Systems*, Long Beach, CA, USA, December.

Su, H., Maji, S., Kalogerakis, E. and Learned-Miller, E. (2015), "Multi-view convolutional neural networks for 3d shape recognition", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Santiago, Chile, December.

Tachella, J., Altmann, Y., Ren, X., McCarthy, A., Buller, G.S., Mclaughlin, S. and Tourneret, J.Y. (2019), "Bayesian 3D reconstruction of complex scenes from single-photon lidar data", *SIAM J. Imaging Sci.*, **12**(1), 521-550. https://doi.org/10.1137/18M1183972

Taunk, K., De, S., Verma, S. and Swetapadma, A. (2019), "A brief review of nearest neighbor algorithm for learning and classification", *Proceedings of the International Conference on Intelligent Computing and Control Systems*, Madurai, India, May.

Ullman, S. (1979), "The interpretation of structure from motion", *Proceedings of the Royal Society of London. Series B. Biological Sciences*, **203**(1153), 405-426. https://doi.org/10.1098/rspb.1979.0006

Van der Jeught, S. and Dirckx, J.J. (2016), "Real-time structured light profilometry: a review", *Opt. Lasers Eng.*, **87**, 18-31. https://doi.org/10.1016/j.optlaseng.2016.01.011

Wan, E., Van Der Merwe, R. and Nelson, A. (1999), "Dual estimation and the unscented transformation", *Proceedings of the International Conference on Neural Information Processing Systems*, Cambridge, MA, USA, November.

Weiss, M. and Baret, F. (2017), "Using 3D point clouds derived from UAV RGB imagery to describe vineyard 3D macro-structure", *Remote Sens.*, **9**(2), p. 111. https://doi.org/10.3390/rs9020111

Westoby, M.J., Brasington, J., Glasser, N.F., Hambrey, M.J. and Reynolds, J.M. (2012), "Structure-from-Motion'photogrammetry: A low-cost, effective tool for geoscience applications", *Geomorphology*, **179**, 300-314. https://doi.org/10.1016/j.geomorph.2012.08.021

Woo, H., Kang, E., Wang, S. and Lee, K.H. (2002), "A new segmentation method for point cloud data", *Int. J. Mach. Tools Manuf.*, **42**(2), 167-178. https://doi.org/10.1016/S0890-6955(01)00120-1

Xu, Y. and Stilla, U. (2021), "Toward building and civil infrastructure reconstruction from point clouds: A review on data and key techniques", *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, **14**, 2857-2885. https://doi.org/10.1109/JSTARS.2021.3060568

Yan, L., Liu, K., Belyaev, E. and Duan, M. (2020), "RTL3D: real-time LIDAR-based 3D object detection with sparse CNN", *IET Comput. Vis.*, **14**(5), 224-232. https://doi.org/10.1049/iet-cvi.2019.0508

Yu, C., Wang, J., Peng, C., Gao, C., Yu, G. and Sang, N. (2018), "BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation", *Proceedings of the European Conference on Computer Vision*, Munich, Germany, September.

Zhen, W., Hu, Y., Yu, H. and Scherer, S. (2020), "Lidar-enhanced structure-from-motion", *Proceedings of the IEEE International Conference on Robotics and Automation*, Paris, France, May.

Zhu, Q., Li, X., Zhao, Y. and Chen, W. (2024), "VSLAM based on deep learning in low-textured scenes", *Measure. Sci. Technol.*, **35**(12), p. 126311. https://doi.org/10.1088/1361-6501/ad7be4