Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

Semantic line framework-based indoor building modeling using backpacked laser scanning point cloud

Cheng Wang [a,b], Shiwei Hou [a], Chenglu Wen [a,*], Zheng Gong [a], Qing Li [a], Xiaotian Sun [a], Jonathan Li [a,b,c]

[a] Fujian Key Laboratory of Sensing and Computing for Smart City and the School of Information Science and Engineering, Xiamen University, Xiamen 361005, China
[b] Fujian Collaborative Innovation Center for Big Data Applications in Governments, Fuzhou 350003, China
[c] GeoSTARS Lab, The Department of Geography and Environmental Management, University of Waterloo, Canada

ARTICLE INFO

Article history:
Received 1 October 2017
Received in revised form 23 March 2018
Accepted 26 March 2018
Available online 25 April 2018

Keywords:
Point clouds
Indoor modeling
Mobile laser scanning
Line framework extraction
Semantic labeling

ABSTRACT

Indoor building models are essential in many indoor applications. These models are composed of the primitives of the buildings, such as the ceilings, floors, walls, windows, and doors, but not the movable objects in the indoor spaces, such as furniture. This paper presents, for indoor environments, a novel semantic line framework-based modeling building method using backpacked laser scanning point cloud data. The proposed method first semantically labels the raw point clouds into the walls, ceiling, floor, and other objects. Then line structures are extracted from the labeled points to achieve an initial description of the building line framework. To optimize the detected line structures caused by furniture occlusion, a conditional Generative Adversarial Nets (cGAN) deep learning model is constructed. The line framework optimization model includes structure completion, extrusion removal, and regularization. The result of optimization is also derived from a quality evaluation of the point cloud. Thus, the data collection and building model representation become a united task-driven loop. The proposed method eventually outputs a semantic line framework model and provides a layout for the interior of the building. Experiments show that the proposed method effectively extracts the line framework from different indoor scenes.

1. Introduction

With the growth of urban populations and the prevalence of large buildings, there is an increasing demand for up-to-date spatial information of indoor environments. Traditionally, 2D floor plans have been regarded as the main source of indoor spatial information. In recent years, 3D modeling and reconstruction of the interior of buildings provide essential 3D models for applications, such as location-based services, building maintenance, disaster rescue, and building renovation planning. The major requirement for these applications is the 3D indoor building models, which are composed of the primitives of the building interiors, such as the ceilings, floors, walls, windows, and doors, but not the objects in the indoor spaces, such as furniture. In this paper, the focus is on the reconstruction of the 3D indoor building model.

Recently, acquiring indoor point clouds has become easier. Popular 3D point cloud measurement systems include stereo cameras, terrestrial laser scanning (TLS), hand-held laser scanning devices, and low-cost depth cameras. Perez-Yus et al. (2016) used RGB-D and fisheye cameras to obtain a scaled 3D model with wide scene reconstruction. Liu et al. (2015) proposed a small set of monocular images of different rooms to form a 3D indoor model using a Markov Random Field model. In addition, to provide 3D data for indoor environments, movable or backpacked systems have been developed based on RGBD cameras or laser scanners (Wen et al., 2014, 2016).

Several methods have been developed for the automated generation of 3D indoor models from point clouds (Jung et al., 2015; Oesau et al., 2014; Ochmann et al., 2014; Xiong et al., 2013; Mura et al., 2014; Wang et al., 2016). Babacan et al. (2016) demonstrated a method to automatically extract floor plans from raw point clouds without using the 3D structure of the indoor environment. Michailidis and Pajarola (2016) presented a method to extract the wall openings (windows and doors) of interior scenes from indoor 3D point clouds. Their method directly extracts windows and doors from a single wall surface. These two methods can be applied only to a single plane surface, such as a floor or wall. Ochmann et al. (2016) developed an automated approach for the reconstruction of parametric 3D building models from indoor point clouds. Armeni et al. (2016) proposed a semantic parsing method for an entire building based on a point cloud acquired by a 3D camera.

* Corresponding author.
E-mail address: clwen@xmu.edu.cn (C. Wen).

https://doi.org/10.1016/j.isprsjprs.2018.03.025
0924-2716/© 2018 International Society for Photogrammetry and Remote Sensing, Inc. (ISPRS). Published by Elsevier B.V. All rights reserved.

However, the following two challenges remain for the task of indoor 3D building modeling:

(1) Data quality challenge. Indoor environments are composed of many independent spaces. The walls between these spaces obstruct vision from one space to another. Thus, 3D data collection must move through different spaces and be measured from different locations. Compared with static TLS and near-ranged RGBD sensors, Simultaneous Localization and Mapping (SLAM)-based mobile laser scanning solutions exhibit better efficiency, range coverage, and geometric consistency (Bosse et al., 2012; Wen et al., 2014, 2016). However, due to the failure of SLAM processing and noises, low-quality sections in SLAM-based laser scanning data are still inevitable. At the same time, heavyocclusions are often on the floor, walls, doors, and windows due to obstacles from furniture.

(2) The challenge of how to effectively represent the indoor model. The indoor building model requires each component to be labeled with its semantic meanings, which will be used in further applications. As primitives of the representation, planes and lines are essential elements for building a 3D model of an indoor scene (Jung et al., 2015; Oesau et al., 2014; Ochmann et al., 2014; Xiong et al., 2013; Mura et al., 2014). However, due to a high level of incompleteness of the point clouds caused by occlusions from furniture, the state-of-the-art methods are ineffective for extracting correct lines and planes. In addition, most existing indoor modeling methods use rule-based prior knowledge or assumptions, such as the ceiling and floor planes should be horizontal, the wall planes should be vertical, etc. In complex cases, these rules may be invalid.

In this paper, we present a novel line framework-based semantic indoor building modeling method using 3D indoor backpacked mapping point clouds in cluttered and occluded indoor environments. Using a self-built backpacked indoor mobile laser scanning system, we effectively and accurately acquire an indoor 3D point cloud. Our proposed method includes three stages: patch-based semantic labeling, 3D line structure feature extraction, and line framework optimization. At the patch-based semantic labeling stage, a trained Conditional Random Fields (CRFs)-based method automatically classifies the raw laser scanning point cloud into four categories: floors, walls, ceilings, and other objects. At the line structure feature extraction stage, 3D line structure features are extracted from labeled point clouds. A semantic Level of Details 3 (LOD3) (Biljecki et al., 2014) building model is obtained by providing a 3D line framework of the walls, ceiling, floor, windows, and open doors. At the line framework optimization stage, a conditional Generative Adversarial Nets (cGAN)-based deep learning model is constructed and applied to the line framework to deal with structure completion, extrusion removal, and line regularization. Also, in this stage, to detect the failure of the SLAM mapping process, the result of line optimization is derived as the quality evaluation of point cloud data.

The main contributions of this paper are as follows:

(1) A SLAM-based backpacked laser scanning system is proposed to collect indoor environment point clouds. With data quality evaluation and filtering, data collection and model representation become a united task-driven loop.

(2) A line framework is proposed to represent the structure of an indoor building model. Our proposed method does not require an assumption of the specific structure of the building, such as that the vertical wall or horizontal floor hypothesis.

(3) A cGAN-based deep learning model is developed to optimize the line framework against clutter background and heavy occlusion in indoor environments.

The pipeline process for our proposed method is shown in Fig. 1.

2. Related works

2.1. Point cloud semantic labeling

In the problem of point cloud labeling, a class label is assigned to each single point or voxel as a label entity in scenes of 3D point clouds. CRFs are often used to propagate contextual information, such as a classification task, between adjacent sites in the field of the computer vision (Schnabel et al., 2007; Moghadam et al., 2013). CRFs usually are based on maximum a posteriori (MAP) learning. To better model correlations in structured data, the Max-Margin Markov Networks (M3N) model (Taskar et al., 2003) has been developed based on max-margin learning. Munoz et al. (2009a, 2009b) used a functional gradient algorithm to learn an Associative Markov Networks (AMNs) (Taskar et al., 2004) model (a type of M3N model) and successfully applied it to 3D point cloud

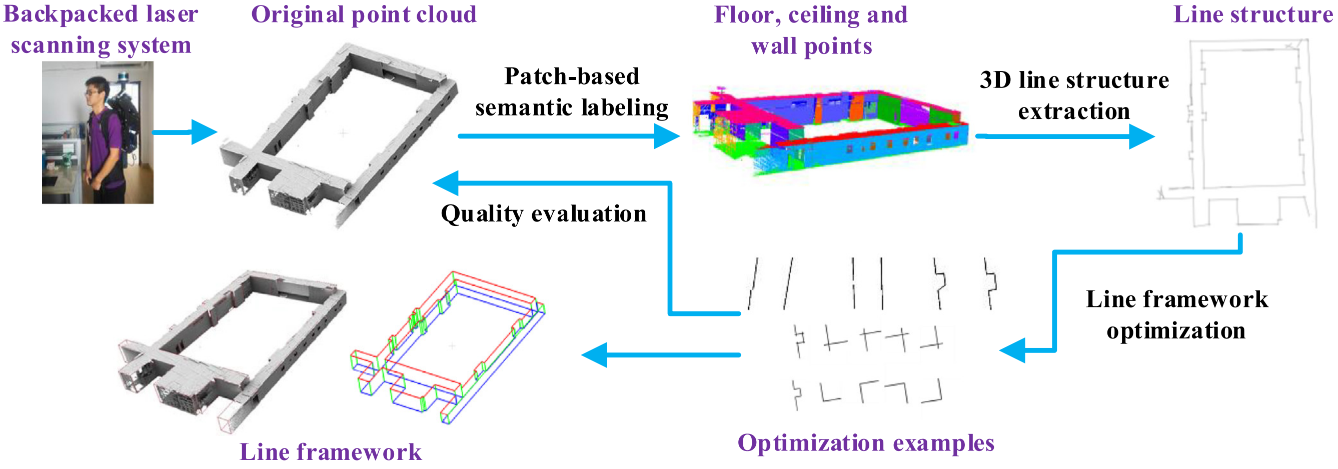
Backpacked laser scanning system
Original point cloud
Patch-based semantic labeling
Floor, ceiling and wall points
3D line structure extraction
Line structure
Quality evaluation
Line framework optimization
Line framework
Optimization examples

Fig. 1. Pipeline of the proposed method.

classification. Moreover, Munoz et al. (2009b) trained a high-order Markov random field (MRF) model to classify 3D point clouds. Shapovalov et al. (2010) proposed applying a non-associative Markov network to classify 3D point cloud data. Luo et al. (2016) extracted 3D patches for colorized point cloud classification, where, in the training process, 3D patches are first extracted from the colorized point cloud data. Then, a 3-D patch-based match graph (3D-PMG) structure is constructed to imply the contextual relationship between various 3D patches.

2.2. Indoor modeling and reconstruction

Current methods for modeling and constructing indoor scenes from point clouds are mainly classified according to 3D line-based, plane fitting-based.

3D point clouds can be represented by a set of 2D images. Extracting 3D lines from 2D planes is much easier than directly extracting 3D lines in 3D point clouds. Jain et al. (2010) proposed a method to extract straight 3D line segments from a set of 2D images. They used a depth value of each single 2D image and then merged the 2D lines from each part for building reconstruction. Lin et al. (2015) proposed a Line-Half-Planes (LHP) model to extract 2D lines by projecting original point clouds onto different image views and then projecting these lines back into 3D space to obtain 3D lines. Lin's method performs well on high density and high accuracy point cloud data; however, it produces excessive noise and details for the indoor laser scanning of point clouds. Oesau et al. (2014) presented a method of permanent structure reconstruction from 3D point clouds. A space-partitioning step, which splits multi-level buildings into several horizontal slices, was first introduced. Then a Hough transform was applied to extract the wall line segments on each horizontal slice projection. The result contains a multi-level structure of an indoor environment, but no windows or doors are involved. This method also assumes that the wall plane is vertical to the floor plane, and the floor and ceiling planes are horizontal.

In an indoor environment, the floor and walls are mostly even planes; therefore, plane fitting-based methods are widely used in indoor modeling and reconstruction based on point cloud data. Sanchez and Zakhor (2012) proposed a model-fitting method based on RANSAC to extract interior planes such as ceilings, floors, and walls from laser scanner point cloud data. Planar regions were detected from point cloud data to extract the plane intersections and corners (Chen and Chen, 2008).

Other techniques, such as Ochmann et al. (2016), developed a parametric modeling approach for the reconstruction of parametric 3D building models from indoor point clouds. Their technique automatically rebuilds a structural model of a multi-room indoor scene. However, this method requires a specific parametric model for a building structure. Furthermore, the method works only for piecewise linear wall structures. The grammar-based modeling method has also been developed for indoor modeling applications (Ikehata et al., 2015). However, this method requires building the modeling grammars manually. We propose in this work to learn from a small number of manually designed examples and automatically created samples. Our proposed indoor modeling method overcomes the shortage of requiring specific parametric models and manually designed grammars.

3. Indoor mobile laser scanning point cloud

3.1. Indoor backpacked laser scanning system

Based on our previous indoor mobile laser scanning systems (Wen et al., 2014, 2016), we built an upgraded backpacked 3D laser scanning system (Fig. 2). Detailed hardware information for the upgraded backpack system is shown in Table 1.

This system contains two 16-beam 3D laser scanners. Each laser scanner consists of sixteen individual laser-detector pairs over the 30° (−15° to +15°) field of view. One laser scanner is placed horizontally to acquire the point cloud P_{Horiz}; the other laser scanner is mounted at 45° below the horizontal one to acquire the point cloud P_{Titl}. Using Eq. (1), we merge the scanners to acquire the global point cloud P_{Global}, where $T_{cali}$is a transform matrix calculated between the two laser scanners (Gong et al., 2018).

$$P_{Global} = P_{Horiz} + T_{cali} * P_{Titl} \tag{1}$$

Table 1
Hardware information.

Equipment	Specifications
Laser scanner	Two 16-beam laser scanners (Velodyne VLP-16)
Battery	lithium battery, 12 V 20 AH, 127 × 72 × 52 mm, 0.973 kg
Processing unit	HP ENVY 15-ae125TX, 384 × 255 × 23 mm, 1.9 kg
Trestle	Carbon fiber, 3.5 kg

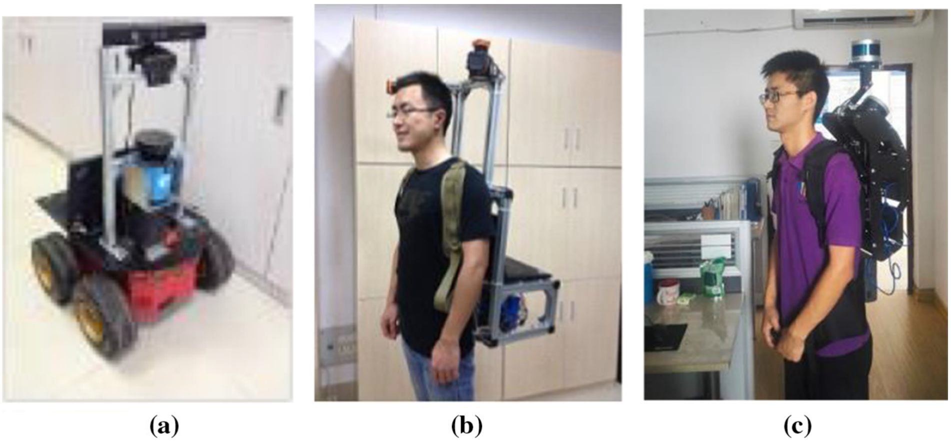

Fig. 2. Indoor mobile laser scanning system. (a) Robot-based mapping system (Wen et al., 2014). (b) Single-beam backpacked laser scanning system (Wen et al., 2016). (c) Multi-beam backpacked laser scanning system.

A human operator, carrying the backpack mapping system, performs the survey at a normal walking speed. To achieve mapping results, the consecutive frames taken by the two laser scanners, from time to time, are registered by the LOAM algorithm (Zhang and Singh, 2014). Compared with our previous systems, the upgraded laser scanning system, by using multi-beam laser scanners and an improved mapping algorithm, acquires indoor 3D point cloud data with higher density and efficiency.

3.2. Characteristics of the indoor point cloud data

Examples of the acquired 3D indoor laser scanning point cloud by our system are shown in Fig. 3 (Data available online: http://www.mi3dmap.net). Point clouds acquired in two complex underground parking areas are given in Fig. 3(a) and (b). Point clouds acquired in a rectangular corridor with a closed-loop are shown in Fig. 3(c). Specifically, Scene 3 consists of two parts with different building heights. The left part of the building is higher than the right part of the building. Point clouds acquired by our system for a multi-room scene are shown in Fig. 3(d). The mapping results indicate that our backpacked laser scanning system provides robust point cloud mapping results for different indoor scenes.

The nature of indoor and outdoor environments is quite different. Compared with an open outdoor environment, an indoor environment is usually complex, narrow, and GNSS-denied. Especially, severe occlusion exists due to the presence of a large amount of furniture. Regarding the nature of the indoor environment, the characteristics of the indoor 3D point clouds acquired by our proposed backpacked laser scanning system are as follows: (1) data incompleteness due to the occlusion and mis-scanned areas caused by narrow space; and (2) data uncertainty due to the cluttered background. Fig. 4 shows the indoor laser scanning point clouds and the corresponding images of two typical indoor scenes. Fig. 4(a) shows simple occlusion by a monitor in a corridor. Fig. 4(b) shows an indoor scene with a high level of occlusion and a cluttered background. As shown in Fig. 4(b), the incompleteness and uncertainty of the data increases dramatically when there is more occlusion and a cluttered background.

4. 3D line framework construction

4.1. Patch-based point cloud semantic labeling

Our proposed method provides prior knowledge of the building data by automatically and semantically labeling the 3D point cloud scenes via learning a labeling model from a certain number of training samples. The labeling task is to assign each 3D point a label from the class set {floor, walls, roof, other objects}. To alleviate the computational burdens associated with labeling a huge number of points, we first extract and describe 3D patches (Luo et al., 2016) from the point clouds and treat them as operating units when labeling.

In the training stage, to yield improved classification results over locally independent classifiers, the learning framework of the AMNs for point cloud labeling is applied to exploit contextual information (Munoz et al., 2009b). A category label $l_k \in \{l_1, \cdots, l_K\}$ is given to the 3D patch, *i*. To describe the 3D

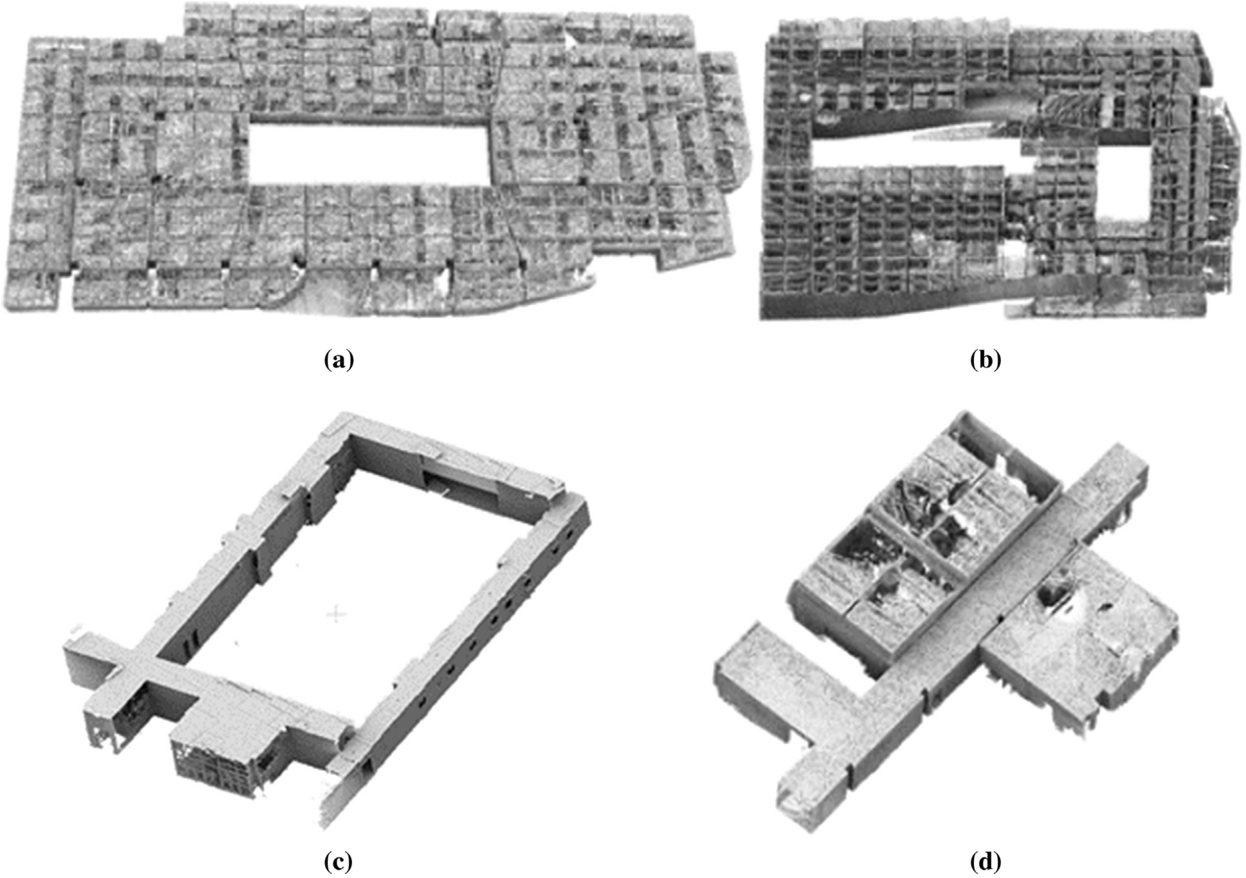

Fig. 3. Examples of 3D point clouds built by our system. (a) Scene 1-Underground parking area. (b) Scene 2-Underground parking area. (c) Scene 3-A closed-loop corridor. (d) Scene 4-Connected rooms.

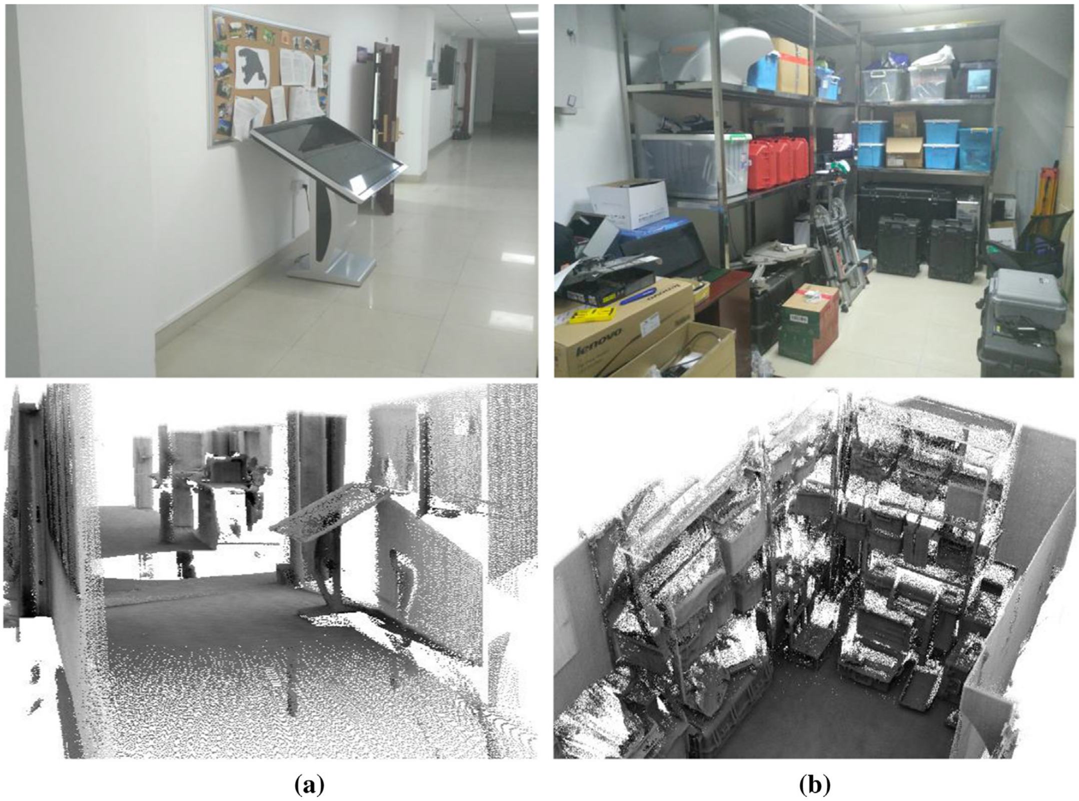

Fig. 4. Typical examples of indoor laser scanning point clouds. (a) Corridor. (b) Room.

patches, Fast Point Feature Histogram (FPFH) descriptors (Rusu et al., 2009) and the height information of the centroid of the points in a patch are computed to form feature vectors $\mathbf{x} = \{x_i, x_{ij}, x_i^c\}$, where $i = 1 \cdots N$, N is the number of 3D patches contained in a point cloud scene; x_i is the feature vector that describes the 3D patch, i, by giving statistics of the local distribution of the points in the 3D patch, i. x_{ij} is the feature vector describing the two spatially adjacent 3D patches i and j; x_i^c is the feature vector describing the clique, c, to which the 3D patch, i, belongs. The assigned labels for the patches are defined as $y = \{y_1, \cdots, y_N\}$. The potential function used in the AMNs model is:

$$\Phi(x, y, W) = \Phi_n(x, y, W_n) + \Phi_e(x, y, W_e) + \Phi_c(x, y, W_c) \tag{2}$$

where Φ_n,Φ_e and Φ_c represents node, edge, and clique potentials, respectively. $W = [W_n, W_e, W_c]$ are the parameters in the AMNs model. Then, we use log-linear potentials to represent the dependence of the node potentials on the extracted features as follows:

$$\log(\Phi_n(x, y, W_n)) = \sum_{i=1}^{N} log(\varnothing_i(y_i^k)) = \sum_{i=1}^{N} W_n^k \cdot x_{\mathrm{i}} \tag{3}$$

where $y_i^k = l_k$ (the label value assigned to node i), and $W_n^k \in A^{d_n}$ are the weights used when a node is assigned to l_k. Similarly, the potential over an edge models an associative/Pott's behavior that favors the two linked nodes taking on the same labels and penalizes as indicated by Eq. (4).

$$\log(\Phi_e(x, y, W_e)) = \sum_{(i,j)\in E}^{N} \log\left(\varnothing_{ij}\left(y_i^k, y_j^o\right)\right)$$

$$\log \varnothing_{ij}\left(y_i^k, y_j^o\right) = \begin{cases} W_e^k \cdot x_{ij}, & l_k \neq l_o \\ 0, & l_k = l_o \end{cases} \tag{4}$$

where $l_k$and $l_o$are the labels of neighboring nodes i and j, and $y_j^o = l_o$. E is the edge set, where each edge is defined by two neighboring nodes. A P$^{\mathrm{n}}$ Pott model (Boykov et al., 2001), which can be efficiently minimized, was used as an energy function. Following this model, the clique potentials$\forall c \in S$, where $S$is the clique set. $E_c(\mathbf{y}_c) = -\log \phi_c(\mathbf{y}_c)$are defined by the high-order energy terms in the AMNs log-liner model, and:

$$\log(\Phi_c(x, y, W_c)) = \sum_{c\in C} \log \phi_c(\mathbf{y}_c)$$

$$\log \phi_c(y_c) = \begin{cases} W_c^k \cdot x^c, & if \forall i \in c, y_i = l_k \\ 0, & otherwise \end{cases} \tag{5}$$

For the AMN learning process, the objective function Eq. (6) is optimized by the subgradient method and graph-cut inference method (Munoz et al., 2009b).

$$\min_w \frac{\lambda}{2} ||W||^2 + \max_y(\Phi(x, y, W) + \zeta(y, \widehat{y})) - \Phi(x, \widehat{y}, W) \tag{6}$$

where $\zeta(y, \widehat{y})$ is a loss function that computes the Hamming distance between the inferred labeling (y) and the true labeling $(\widehat{y})$. λ is a regularization term.

For the AMN inference process, inferring category labels from unlabeled scenes is carried out in the labeling stage. The category labels, $\mathbf{y}^*$, for 3D patches are estimated effectively by maximizing Eq. (7) via the α-expansion graph-cut method (Boykov et al., 2001).

$$\mathbf{y}^{*} = \arg\max_{y} P_{w}(y|x) = \arg\max_{y} (\Phi(x, y, W)) \tag{7}$$

For the four categories of point clouds obtained, only the floor, ceiling, and wall points are used in the further indoor modeling.

4.2. Line structure extraction on 3D point cloud

To minimize the effects from occlusion on line extraction, a line structure extraction method is developed directly on the 3D point cloud. We first extract a flat (but not specifically horizontal) boundary line using floor and ceiling points obtained from the above labeling results. However, due to the complexity of the indoor environment, the labeled floor and ceiling planes can be highly incomplete. In this situation, we merge the floor and ceiling points to obtain a more complete flat plane only if the floor and ceiling planes are relatively parallel. If the floor and ceiling planes are not parallel, line extraction is applied directly to each plane.

The first step of this method is to calculate the normal vector, $\vec{n_i}$, of every single point p_i. We choose the dimensional coordinate system origin $\mathrm{O}(0, 0, 0)$ as the starting viewpoint. To calculate the normal vector $\vec{n_i}$ of point p_i, we select the k points nearest to point p_i as one plane, and the normal vector of the plane is taken as the normal vector of the point. Too many numbers of k may result in lower accuracy of the calculated normal vector. Too few numbers of k result in a large computational cost and may result in a greater number of errors included during the calculation. In the method, we set the value of k at 35 for our data.

Then, the tangent plane of each point is calculated. For each input point, p_i, of the original point cloud P, the tangent plane $T_{p_i}$is expressed by its center pointo_i, and the normal vector, $\vec{n_i}$, is as follows:

$$T_{p_i} = (\mathrm{o_i}, \vec{n_i}) \tag{8}$$

The Euclidean distance from any point, p_i, to T_{p_i} in 3D space is calculated as follows:

$$\mathrm{dist}(p_i, T_{p_i}) = \left|(p_i - \mathrm{o_i}) * \vec{n_i}\right| \tag{9}$$

For the set of K neighborhoods of p_i: $B_k(p_i)$, the best fitting plane is obtained by solving the following equation:

$$\underset{T_{p_i}}{\mathrm{argmin}} \sum_{p_i \in B_k(p_i)} \mathrm{dist}(p_i, T_{p_i})^2 \tag{10}$$

Because all tangent planes, T_{p_i}, for all p_i have been computed, we randomly select one point from the original point cloud as a seed point. Next, we select one point, p_i, and create a new facet, $f_i = \left(\{K_i^m\}, x_i, \vec{T_i}\right)$ for the seed point, p_i, where $\vec{T_i}$ is the unit normal vector of T_{p_i}. Then, an improved region growing procedure commencing with f_i is carried out. We add each adjacent point, p_j, to the facet, f_i, if p_j is not used until now and satisfies the following three criteria: (i) the angle between $\vec{T_i}$ and $\vec{T_j}$ does not exceed the tolerance θ, (ii) the distance from $p_j$to p_i does not exceed R_{seed}, and (iii) the orthogonal distance from p_j to f_i is smaller than $\sigma/2$. Here, R_{seed} is used to constrain the radius of the facets to ensure that the large facets can be segmented into smaller pieces.

When the facet f_i is determined, we choose another point p_k from among the last points of the original points and repeat the steps to find another facet f_k. This iterative process is performed until most of the points have been divided into different facets as shown in Algorithm 1:

Algorithm 1. Generating Facets

```
Input: Original point set P, point to facet distance threshold $\sigma$,
   the tolerance angle $\theta$ between tangent planes, the number of
   neighboring points (k)
Output: facets set F
For each $p_i$ In P Do
  Calculate $T_{p_i}$using Eq. (10)
End For
Used(P) ← false
For each $p_i$ In P Do
  If Used($p_i$) ≠ false Continue
  $f_i \leftarrow \left(\{K_i^m\}, x_i, \vec{T_i}\right)$
  For each $p_j$In P Do
  If Used($p_i$) ≠ false Continue
    If $\left\{\begin{array}{l} angle(T_i, T_j) < \theta \\ dist(p_i, f_i) < \sigma/2 \\ dist(p_i, p_j) < R_{seed} \end{array}\right.$ Then
      $f_i \leftarrow f_i \cup p_j$
      Used($p_j$) ← true
    End If
  End For
End For
```

We then extract line segments from these facets. Inspired by Lin's work (Lin et al., 2015, 2017), we improved their work to make it suitable for indoor point cloud data. Lin's original method works specifically for extracting the lines of building exteriors from high resolution and high accuracy point clouds. To apply Lin's method to relatively low resolution and low accuracy indoor laser scanning point clouds, we first increase the area of the facets (the parameter, R_{seed}, mentioned above) to reduce the number of lines in the internal plane and maintain as long a border line as possible. Second, we decrease the angle, θ, between two tangent planes to separate different tangent planes better. Last, we generate a continuous head-to-tail straight line to approximate the edge of the curve structure. In Lin's method, several discontinuous straight lines with gaps are generated in this stage.

To obtain positive line results, the first step is to extract the boundary points of each facet, f_i. The vertices of the α-shape of f_i are presented as the boundary points of the f_i. However, these boundary points can also contain the intersecting points of two adjacent coplanar facets. To overcome this problem, we define a facet, F_i of f_i, which contains adjacent coplanar facets and the current facet, f_i. Then we extract the α-shape points of F_i and compute the desired intersecting points, P_i.

The next step is to obtain a line segment of F_i based on the boundary points, P_i. Because of the relatively low-cost single/multi-beam laser scanner used in data acquisition, indoor mobile laser scanning point clouds have the characteristics of relatively low density and limited precision, which results in messy extracted lines and multiple close parallel lines. Rather than directly group the boundary points into line segments, we group the boundary points into a cylinder to filter false detections by a cylinder-based alignment method (Lin et al., 2017). To reduce line extraction false positives and ensure a good line segment result, we extended the Number of False Alarms (NFA) algorithm (Desolneux et al., 2000; Von Gioi et al., 2010) to 3D and kept only one line for each cylinder.

4.3. Wall opening detection and line framework formation

The wall line results are different from the floor and ceiling line results. We retain as many floor or ceiling lines as possible. Because the wall borderlines have been extracted already from the floor and ceiling point clouds, the wall line results require only the door and window lines. Therefore, we drop the wall borderlines and retain only the internal lines from the windows and doors.

For indoor scenes, the windows and doors are mostly rectangular. Only two intersecting edges are necessary to determine a rectangle. As for the original wall line extraction results, a k-means method is applied to capture potential door and window lines. For each wall plane, to find the best line extraction result, a different k value is set from 0 to 9. When the potential door or window lines are obtained, the longest line length is calculated to determine if it belongs to a door or a window. In this step, the detection results are refined using the hypothesis that doors and windows are rectangular. The last step is line framework formation. Because 3D line structures are extracted in 3D space, we obtain the line framework result by combining all the line structures extracted from floors, ceilings, and walls.

5. Line framework optimization using deep learning model

5.1. Problems of line structure extraction

The results obtained using the line extraction framework, presented in Section 4, are usually imperfect because of the occlusion and cluttered background (Fig. 5). The problems of line structure extraction are summarized as follows: (1) Irregular structures (a parallel or orthogonal relationship between some lines) are due to the uncertainty and noise level in the data (Fig. 5(a)). (2) Incomplete structures and disconnected lines exist because of the occlusion (Fig. 5(b)). (3) Extrusions remain because of the uncertainty and noise in the data (Fig. 5(c)).

Usually, a line regularization step, commonly using the rule-based method, is required after line extraction. The general line framework of a building meets some specific building rules, and lines can be further refined by regularization based on these rules. However, the above-mentioned rule-based line regularization method requires a large amount of human interaction and depends on pre-defined assumptions. Likewise, in complex indoor environments, fixed rules often become invalid.

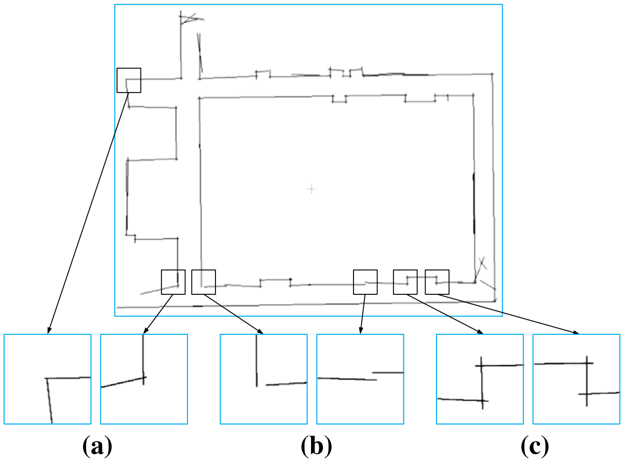

Fig. 5. Examples of imperfect and incomplete line structure extraction.

5.2. CGAN deep learning model

To remove the extrusions, to complete and to regularize the line structure, we introduce a conditional Generative Adversarial Nets (cGAN)-based deep learning model to optimize the imperfect line framework. The GAN (Goodfellow et al., 2014) model trains a generator, *G*, to produce outputs that are indistinguishable from the "real" sample, and trains a discriminator, *D*, to distinguish the outputs of the generator as much as possible. The cGAN model (Isola et al., 2017), an improved network of the GAN model, originally completes the translation between semantic labels and photos, architectural labels and photos, edges and photos, etc. In this paper, we innovatively apply the cGAN model to optimize the line structure.

A GAN learns the mapping from the random noise vector, *z*, to output, *y*; cGAN learns a mapping from input, *x*, and random noise vector, *z*, to output, *y*. The same as GAN, the cGAN model learns a loss function automatically to satisfy different tasks without designing a new loss function. cGAN uses an objective function as follows:

$$G^* = arg\min_{G}\max_{D}\mathcal{L}_{cGAN}(G,D) + \lambda\mathcal{L}_{L1}(G) \tag{11}$$

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y\sim pdata(x,y),z\sim p_z(Z)}[||y - G(x,z)||_1] \tag{12}$$

By introducing a bound term, *L1* distance, the outputs of generator, *D*, are not only similar to the real sample, but also more closely related to the input conditional sample. In generator architecture, a U-net encoder-decoder network is adapted because that U-net makes better use of the low-level information (Ronneberger et al., 2015). A *Patch*GAN (Isola et al., 2017) is designed as the new discriminator architecture to improve the efficiency of the discriminator. This new discriminator architecture attempts to classify whether each $N \times N$ patch in an image is real or fake, then averages all responses to provide the ultimate output of *D*.

5.3. Line framework optimization using cGAN model

Since the cGAN model is working in a 2D plane, all the line structures extracted from each point cloud category (floor, wall, and ceiling) are first projected onto their own planes. To project each point, the coordinate system of every point is transformed from the previous *oxyz* coordinate system to a new *oxyz* coordinate system. In the new coordinate system, to achieve projection, the *z* coordinate of each point is set to zero. The detailed steps to project are as follows: Firstly, a point $o(x_0, y_0, z_0)$ is randomly chosen in the plane as the new origin. Then, two orthogonal unit vectors $u_x = (u_{x1}, u_{x2}, u_{x3})$ and $u_y = \left(u_{y1}, u_{y2}, u_{y3}\right)$ are chosen in the plane as the new *x* axis and the new *y* axis; the starting point of these vectors is *o*. Next, a unit vector $u_z = (u_{z1}, u_{z2}, u_{z3})$ is chosen as the new *z* axis from the normal vector of the plane; the starting point of this vector is also *o*. Finally, a translation matrix $T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x_0 & -y_0 & -z_0 & 1 \end{bmatrix}$ and a rotation matrix $R = \begin{bmatrix} u_{x1} & u_{x2} & u_{x3} & 0 \\ u_{y1} & u_{y2} & u_{y3} & 0 \\ u_{z1} & u_{z2} & u_{z3} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ are obtained resulting in new coordinates for each point as follows:

$$(x, y, z, 1) = (x, y, z, 1) \cdot T \cdot R \tag{13}$$

After obtaining the projection, *x*and *y*are converted into rows and columns in a 2D image. The 2D image is divided into several 256×256 sub-images. These sub-images are classified by a

VGG-16 convolutional neural network (Simonyan and Zisserman, 2014) to extract features and use three full connected layers to classify the features. The convolution layers use 3 ×3 kernel and add batch normalization. Max-pooling is used to do down-sampling. During the training, 2000 training samples were used. The batch size and epoch were set to be 32 and 200, respectively. The results are the input to different cGAN models (see Fig. 6). After obtaining the optimized 2D lines by cGAN models, the pixels on the optimized 2D lines are transformed back to 3D points by Eq. (13). Finally, the 3D points are fitted to the 3D lines by the linear least squares fitting algorithm.

During this process, the main precision loss comes from the process of projecting 3D lines onto 2D lines. The original 3D lines are vectors in 3D space, and the 2D lines on the image are several sets of discrete pixels. Different pixels per unit length of a 3D line results in different precision loss. In general, the more pixels per unit length of a 3D line converted, the smaller the resulting precision loss. A precision loss comparison for Scene 1 and Scene 3 with 50, 100 and 200 pixels per meter of a 3D line is given in Table 2. The average distance from all projected 3D points onto the original 3D lines is used to measure the precision loss. In this paper, taking into consideration both computational cost and precision loss, a meter of the 3D line is converted into 200 pixels. The average precision loss is about 1 mm. The results indicate that the impact of the precision loss during projection is relatively small.

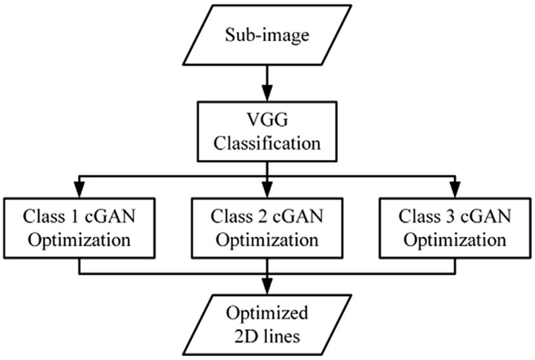
Sub-image
VGG Classification
Class 1 cGAN Optimization
Class 2 cGAN Optimization
Class 3 cGAN Optimization
Optimized 2D lines

Fig. 6. The flow chart of processing 2D data.

Table 2
Precision loss comparison.

Pixels per meter of a 3D line	Average distance (Scene1)	Average distance (Scene3)
50	5.13 mm	4.81 mm
100	2.61 mm	2.56 mm
200	1.34 mm	1.17 mm

To complete the structure optimization task, we construct three cGAN modules: structure completion, extrusion removal, and line regularization modules (see Fig. 7). Each module only deals with the imperfections in each data. The generator is a symmetrical fully convolutional network containing 16 convolutional layers with 4 × 4 convolution kernels. The first eight layers of the generator form an encoder; the second eight layers form a decoder. The discriminator network consists of four convolutional layers; the final layer outputs the discrimination result by a sigmoid activation function.

All training samples are divided into three categories to meet different optimization cases. Training data of the structure completion module consists of samples with two disconnected lines and the corresponding connected lines. The training data of the extrusion removal module consists of samples with extra parts and the corresponding samples without extra parts. The training data for line regularization consist of the samples with unparallel or non-orthogonal lines and the corresponding samples after regularization. To ensure sufficient training samples for the cGAN deep learning model, we prepared the training samples in two ways. A small number of training samples is manually created by cutting from the rule-based manual regularization results. A large set of training samples is generated by a computer program using manually developed rules-related building principles. Some examples of these rules are: an indoor structure should be a closed structure; the boundary of an indoor structure is unique; indoor framework lines should be continuous; building interior framework lines are either parallel or orthogonal.

Specifically, this framework optimization method can be further applied to other building structures by easily replacing the training samples generated from the new building structure. Meanwhile, since the imperfections of the data are detected automatically by the cGAN models, the data requires optimization at the same time. The data requiring optimization usually refers to the point cloud data with low quality. Meanwhile, the quality of the indoor laser

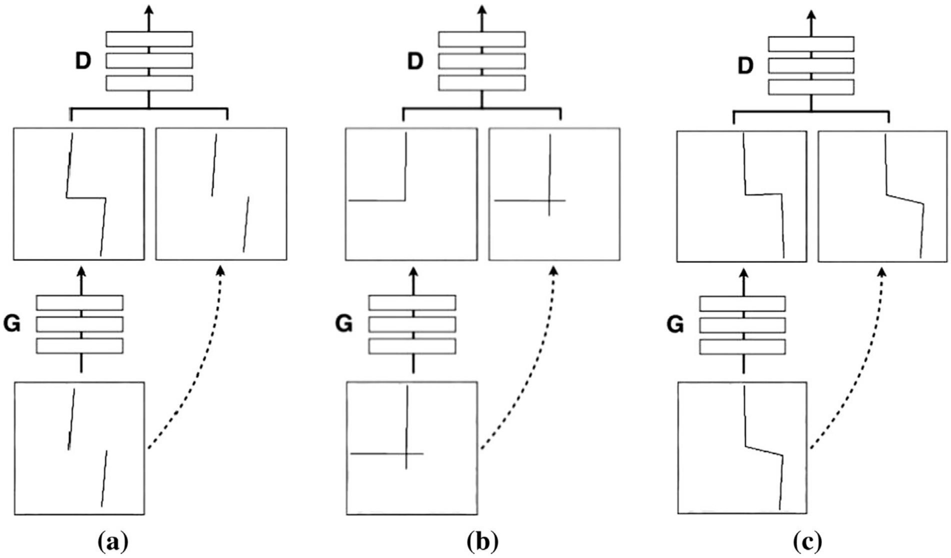

Fig. 7. Three cGAN modules for framework structure optimization. (a) Structure completion module. (b) Extrusion removal module. (c) Line regularization module.

scanning data is affected directly by the mapping process. For example, the varied walking speed of the human operator may result in inconsistent point cloud density. A too fast turning angle of the system may result in the failure of the SLAM process and eventually lead to mis-registration of point cloud frames. With these references, we can trace back to the original mapping process and assess the data quality, as well as filter the low-quality point cloud data. With the close looped data quality evaluation and filtering, the data collection and model representation become a united task-driven loop.

6. Experiments and results

6.1. Patch-Based point cloud semantic labeling

The labeling model was trained on labeled point clouds extracted from different indoor 3D scenarios. The learning parameters were determined by classifying validation point clouds. The training data consists of more than 300,000 points. The class label for each 3D point in the training samples was manually selected and labeled for learning parameters in AMNs. Some examples of training samples are shown in Fig. 8.

Labeling results for Scene 1 and Scene 2 are given in Figs. 9 and 10, respectively. The original point cloud of the parking area (Fig. 9(a)) is labeled by the proposed labeling method (Fig. 9(b)). To better demonstrate the labeling results, we provide each category of point cloud data separately in Fig. 9(c). The segmented point clouds are labeled into ground (blue), wall (green), ceiling (red), and others (yellow).

To quantitatively assess the accuracy and correctness of the semantic labeling results on a test dataset, we selected the following three measures: Precision, Recall, and F1-measure. Precision describes the percentage of true positives in the ground truth. Recall depicts the percentage of true positives in the semantic labeling results. F1-measure is an overall measure. The three measures are calculated on points as follows:

$$\text{precision} = \frac{TP}{TP + FN} \tag{14}$$

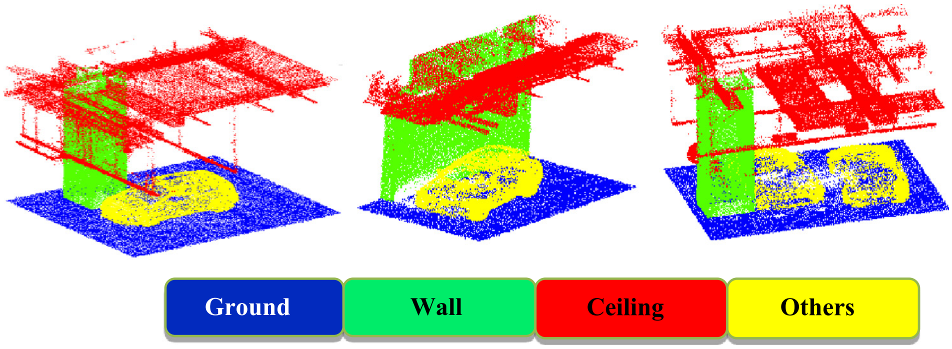

Fig. 8. Examples of training samples. Different colors represent different categories. Blue points represent the floor, green points represent the wall, red points represent the ceiling, and yellow points represent other objects. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

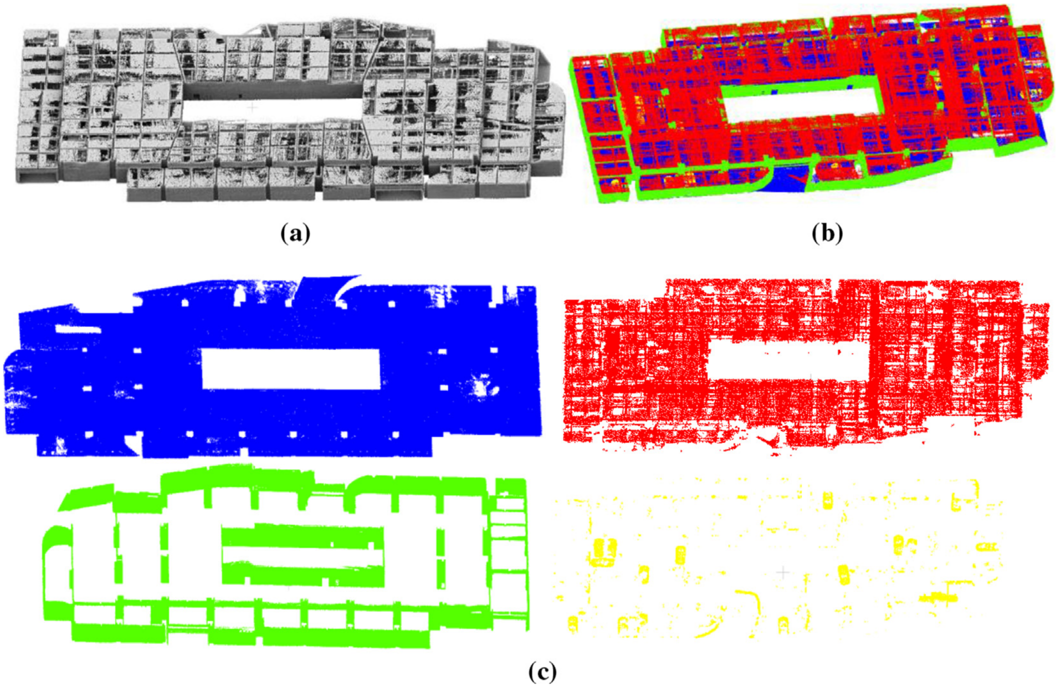

Fig. 9. Labeling results of indoor Scene 1: (a) Original point cloud. (b) Semantic labeling results. (c) Labeling results for different categories: ground (blue), wall (green), ceiling (red), others (yellow). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

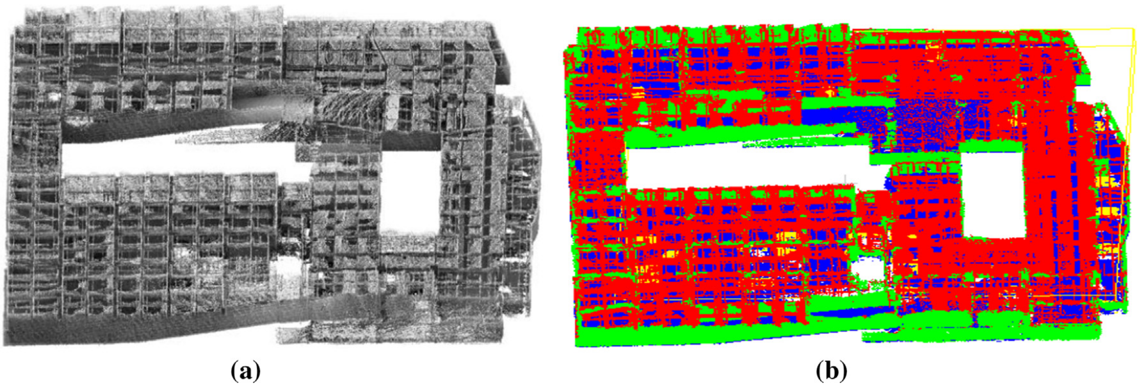

Fig. 10. Labeling results of indoor Scene 2: (a) Original point cloud. (b) Semantic labeling results. Color code: ground (blue), wall (green), ceiling (red), others (yellow). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$$\text{recall} = \frac{TP}{TP + FP} \tag{15}$$

$$F_{1-measure} = \frac{2 * precision * recall}{precision + recall} \tag{16}$$

where TP, FN, and FP represent the number of true positives, false negatives, and false positives, respectively. The quantitative evaluation results using these three measures of indoor Scene 1 and indoor Scene 2 are shown in Tables 3 and 4, respectively. The proposed semantic labeling method achieved, in labeling Scene 1, an average precision, recall, and $F_{1-measure}$ values of 0.86, 0.89, and 0.87, respectively, and, in Scene 20.90, 0.84 and 0.87, respectively.

6.2. Line framework construction

We take Scene 3 as an example scene to show the detailed line extraction results by the proposed line structure extraction method. In this experiment, we set some parameters to be constant (see Table 5). As we mentioned in Section 4.2, θ is the tolerance between the tangent planes, σ is the orthogonal Euclidean distance from point to tangent plane. Lines are discarded if their lengths are smaller than the length-threshold. Min clusters is the minimal

Table 5
Important parameters used in algorithm.

θ(Â°)	σ(m)	Length-threshold(m)	Min clusters (number)
10	0.2	0.1	30

Table 6
Different R_{seed} and K values *VS.* different line number extracted.

Scene 3	R_{seed}(m)	K(number)	Line number
Ground-plane	1.0	15	338
Ground-plane	1.0	30	235
Ground-plane	3.0	15	292
Ground-plane	3.0	30	243
Ceiling-plane1	1.0	15	245
Ceiling-plane1	1.0	30	168
Ceiling-plane1	3.0	15	232
Ceiling-plane1	3.0	30	158
Wall-plane	1.0	15	81
Wall-plane	1.0	30	79
Wall-plane	3.0	15	43
Wall-plane	3.0	30	40

Table 3
Labeling confusion matrix for Scene 1.

		Inferred label				Recall
		Floor	Wall	Ceiling	Others	
True Label	Floor	152,025	556	3459	362	0.97
	Wall	2361	124,452	8801	3311	0.91
	Ceiling	9885	11,274	222,147	3242	0.90
	Others	766	145	3444	14,170	0.77
Precision		0.92	0.91	0.93	0.67	

Table 4
Labeling confusion matrix for Scene 2.

		Inferred label				Recall
		Floor	Wall	Ceiling	Others	
True Label	Floor	363,190	1079	28,176	401	0.93
	Wall	3421	322,084	29,145	1284	0.91
	Ceiling	11,762	18,899	401,651	1750	0.93
	Others	455	5338	6675	16,230	0.57
Precision		0.96	0.93	0.86	0.83	

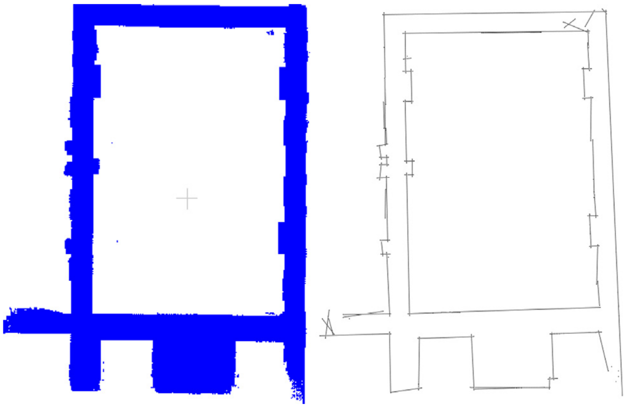

Fig. 11. Line extraction results on floor point cloud.

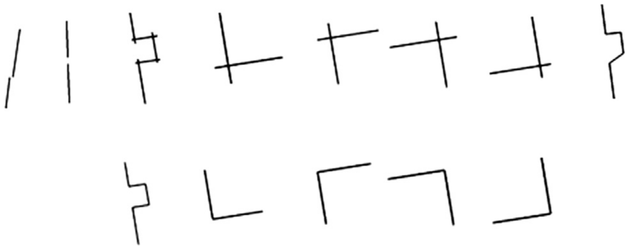

Fig. 14. Examples of training samples. On top are the input samples, and on the bottom are the target samples.

number of points in the tangent plane. The tangent plane is abandoned if it has a fewer number of points than min clusters.

The main parameters affecting the line extraction results are R_{seed} and K values. R_{seed} is used to determine the size of a facet; K

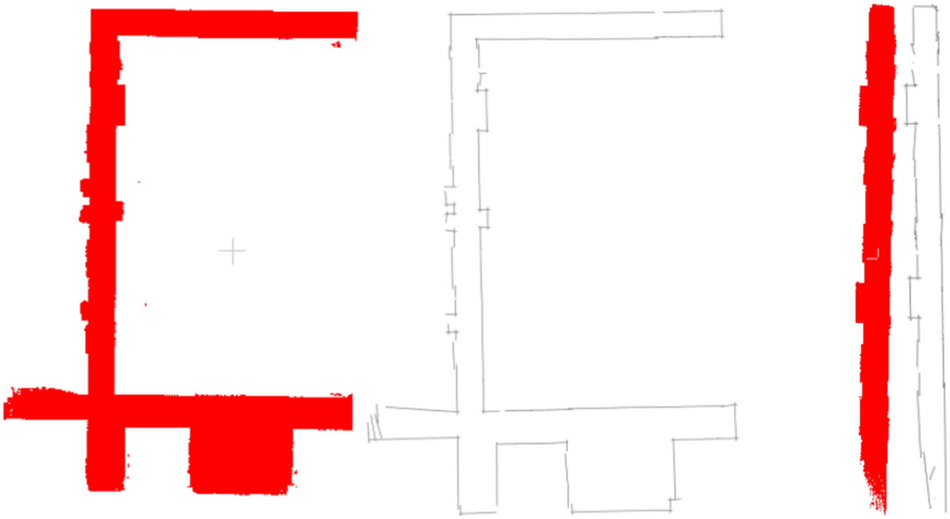

Fig. 12. Line extraction results on ceiling point cloud 1 (first two) and ceiling point cloud 2 (latter two).

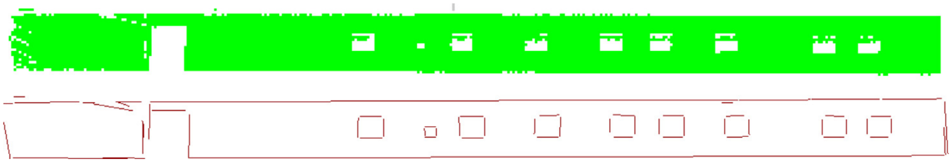

Fig. 13. Line extraction results on wall point cloud.

Table 7
Number of line extracted and running time for each category point cloud of Scene 3.

Description	Number of points	Number of lines	Line segmentation (s)	Line extraction (s)	Total running time (s)
Ground plane	0.92 million	424	38.35	29.296	67.65
Floor plane 1	0.69 million	351	31.45	18.044	49.49
Floor plan 2	0.23 million	117	23.91	6.341	30.25
Wall plan	0.06 million	79	12.93	1.541	14.47

Table 8
Number of line extracted and running time for different scenes.

Description	Number of Points	Number of lines	Line segmentation (s)	Line extraction (s)	Total running time (s)
Scene 1	7.90 million	2652	292.19	84.88	377.07
Scene 2	3.85 million	1819	166.44	62.73	229.17
Scene 3	2.10 million	1652	114.66	69.99	184.66
Scene 4	8.62 million	1289	456.74	87.64	544.38

Table 9
Testing sample size, time and average non-overlapping percentage.

Module	Testing sample size	Time (s)	Average non-overlapping percentage (%)
Structure completion	500	23.34	0.08
Extrusion removal	500	23.09	0.17
Line regularization	500	23.38	0.52

is used to filter the isolated points. As shown in Table 6, different R_{seed} and K values lead to a different number of lines extracted. Also indicated in Table 6 is the larger the $[R_{seed}, K]$ value, the fewer number of lines extracted from floor, ceiling, and wall point clouds. However, $[R_{seed}, K]$ should not be so large that some important missing lines are avoided. Considering the above situations, we set $[R_{seed}, K] = [3.5, 35]$ in the experimental data to obtain the final results. The line extraction results on floor, ceiling, and wall point clouds of Scene 3 are shown in Figs. 11, 12, and 13, respectively.

In addition, Table 7 provides the runtime of line extraction for each point cloud category in Scene 3. Our code runs on a Windows 10, CPU Inter® Core™ i5-4460@3.20 GHz, 12G RAM. We found that the larger number of points required longer running times. For example, with the point number of 0.92 million, the total running time is about 67 s. Meanwhile, the running time of the line segmentation step is more than two times the running time of the line extraction step. The number of lines and the running time are also provided for Scenes 1, 2, 3, and 4 in Table 8. The greater the number of points, the greater the number of lines extracted, and the total running time is longer.

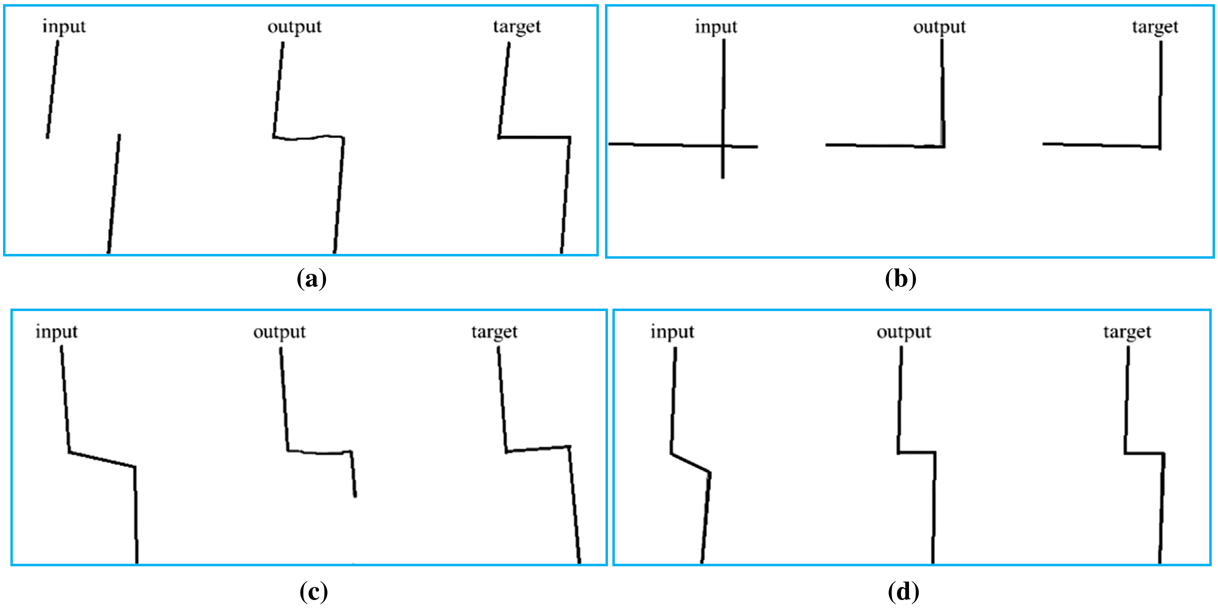

Fig. 15. Examples of testing results. (a) Structure completion. (b) Extrusion removal. (c) and (d) Line regularization.

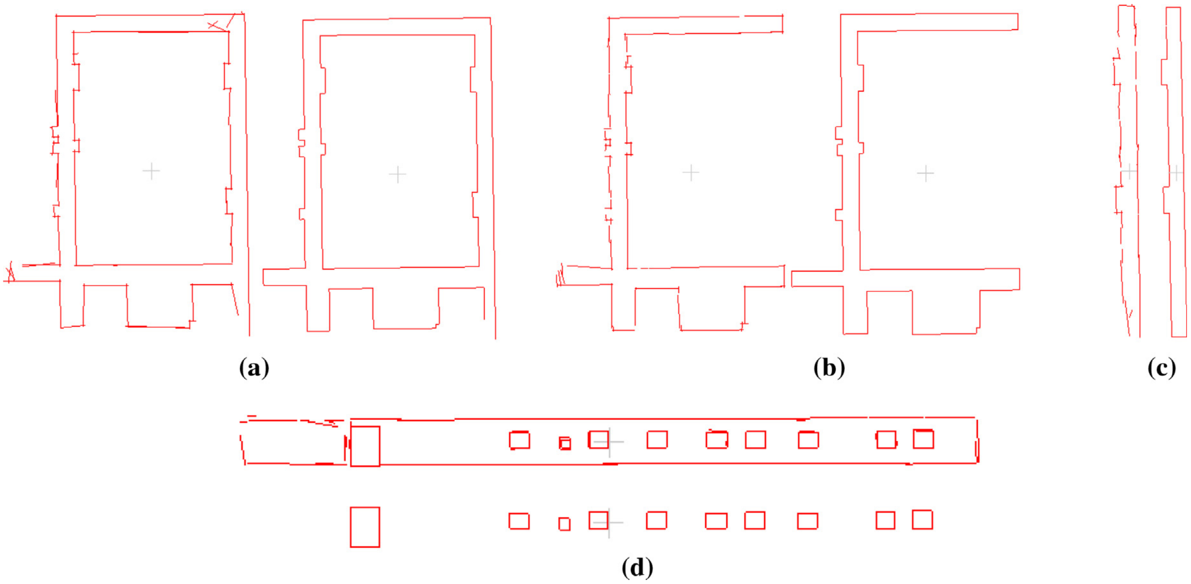

Fig. 16. The before and after framework optimization of lines extracted from Scene 3. (a) Floor lines. (b) and (c) Ceiling lines. (d) Door and window lines.

6.3. Line framework optimization

To optimize the line framework, the line structures extracted are first carefully studied. Besides the real line extraction results, the training data are automatically generated by a computer program. The training data falls into three categories:

(1) Structure completion: We first generated two disconnected lines in a 256×256 pixel size image as input samples, then we connected them as target samples.
(2) Extrusion removal: We first generated a corner as target samples (ground truth), then we lengthened two lines to obtain extrusion parts as input samples.
(3) Line regularization: We first generated some lines, which are unparalleled or non-orthogonal as input samples, and then we adjusted them to be parallel or orthogonal as target samples.

For each category, there are 1000 samples in a training set. During the training, the batch size and epoch of the cGAN model were

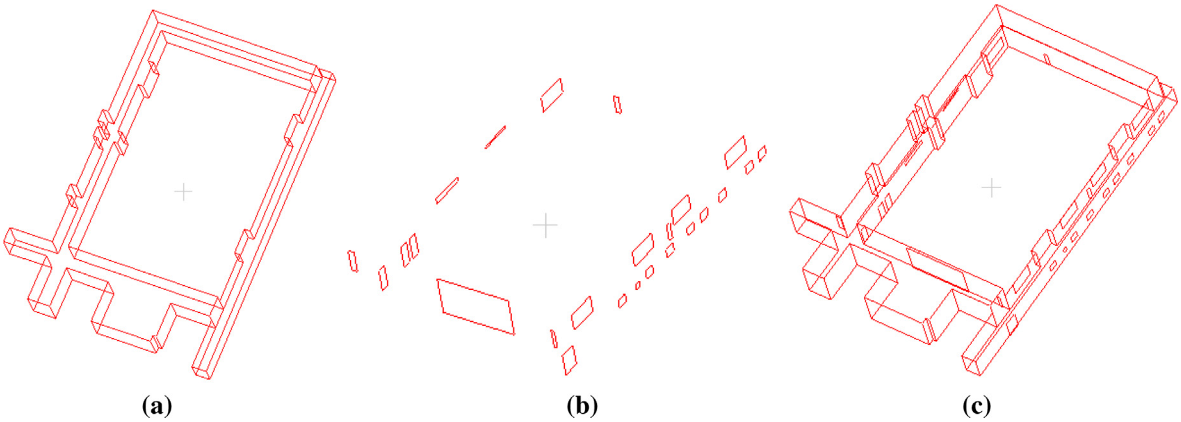

Fig. 17. Line framework after optimization. (a) Floor, ceiling, and wall lines. (b) Windows and doors. (c) Combined line framework.

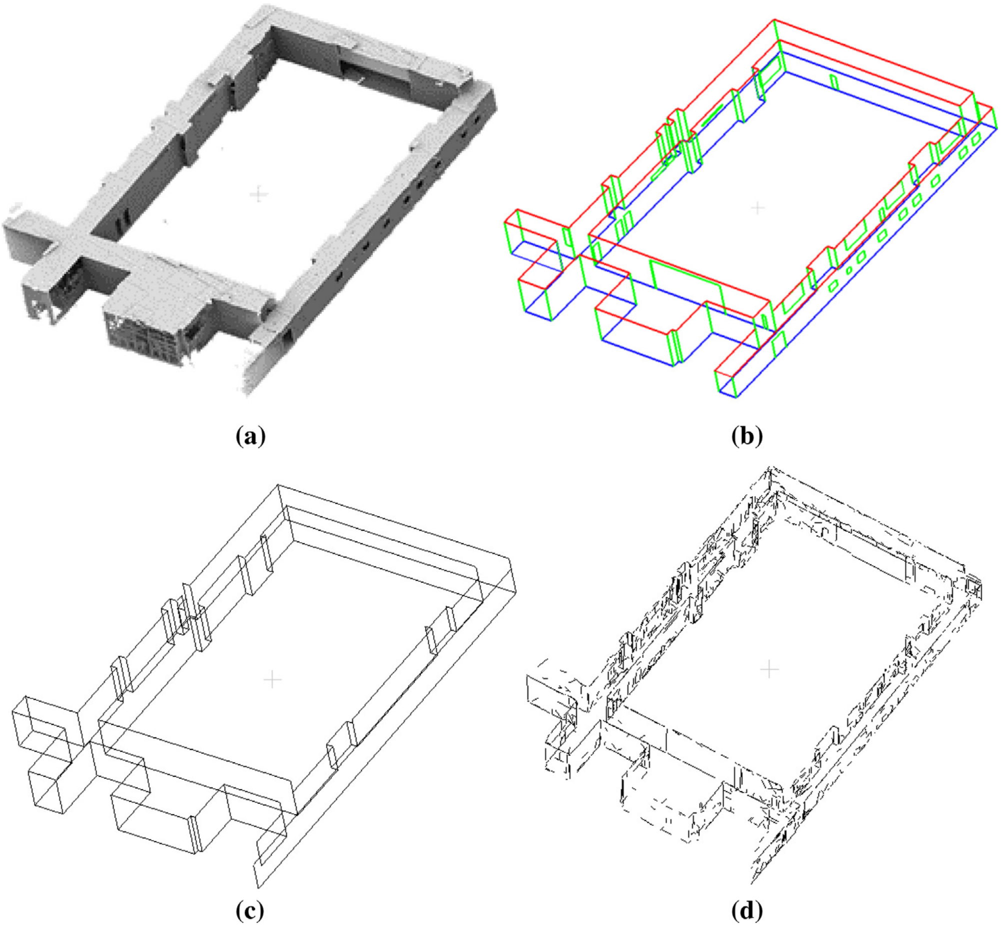

Fig. 18. Result comparison on Scene 3. (a) Raw point cloud. (b) Our method. (c) Oesau's method. (d) Lin's method.

set to be 4 and 500, respectively. The average training time for each category ranged from six to eight hours on two Nvidia Titan X GPUs. Some examples of training samples are shown in Fig. 14.

The testing sample size and time consumption are shown in Table 9. For each category, 500 samples are used for testing. The running time for each category is about 23 s. In addition, based on the principle of rule-base regularization, the target sample results were also provided manually. Then, we compared the generated target results to the manual target results point by point and counted the number of points overlapping in the two results. Lastly, we used the proportion of non-overlapping points to evaluate the performance of the proposed cGAN-based line framework optimization method.

Average percentages of non-overlapping points of each category from 500 testing samples are also given in Table 9. Overall, the structure completion module has the lowest average non-overlapping percentage of 0.08%; the extrusion removal module has an average non-overlapping percentage of 0.17%, and the line regularization module has the highest average non-overlapping percentage of 0.52%. The results are understandable, because, even for a human operator, it is more difficult to regularize a line structure than to complete a line and remove the extrusions. In general, the average non-overlapping percentages of the three modules are all less than 1%, which indicates the cGAN model is effective for line framework optimization.

To further analyze the performance of the proposed framework optimization method, we give some challenging testing results for each category (see Fig. 15). For example, for structure completion, even though the distance between two lines is relatively long, it can still be used to connect the lines (Fig. 15(a)). Since line regularization is a more difficult task, we give a fail testing example in Fig. 15(c). As shown in the figure, the line is partially regularized, but a part of the line remains missing.

After optimizing the line framework, we obtain a better indoor framework structure. The before and after framework optimization of floor lines extracted from Scene 3 are shown in Fig. 16(a). The before and after framework optimizations of ceiling lines extracted from Scene 3 are shown in Fig. 16(b) and (c). The before and after framework optimization of window and door lines extracted from Scene 3 are shown in Fig. 16(d). Shown in Fig. 17 are the combined line framework optimization results, which indicate that the proposed line framework optimization achieves good results.

6.4. Comparison and more testing results

We compared our method with Lin's (Lin et al., 2017) and Oesau's (Oesau et al., 2014) methods (Figs. 18 and 19).

Our method vs. Lin's method. Lin's method provides too many detailed lines and requires a line-refining process (Figs. 18(d) and 19(d)). Our method borrows the basic idea of Lin's method to

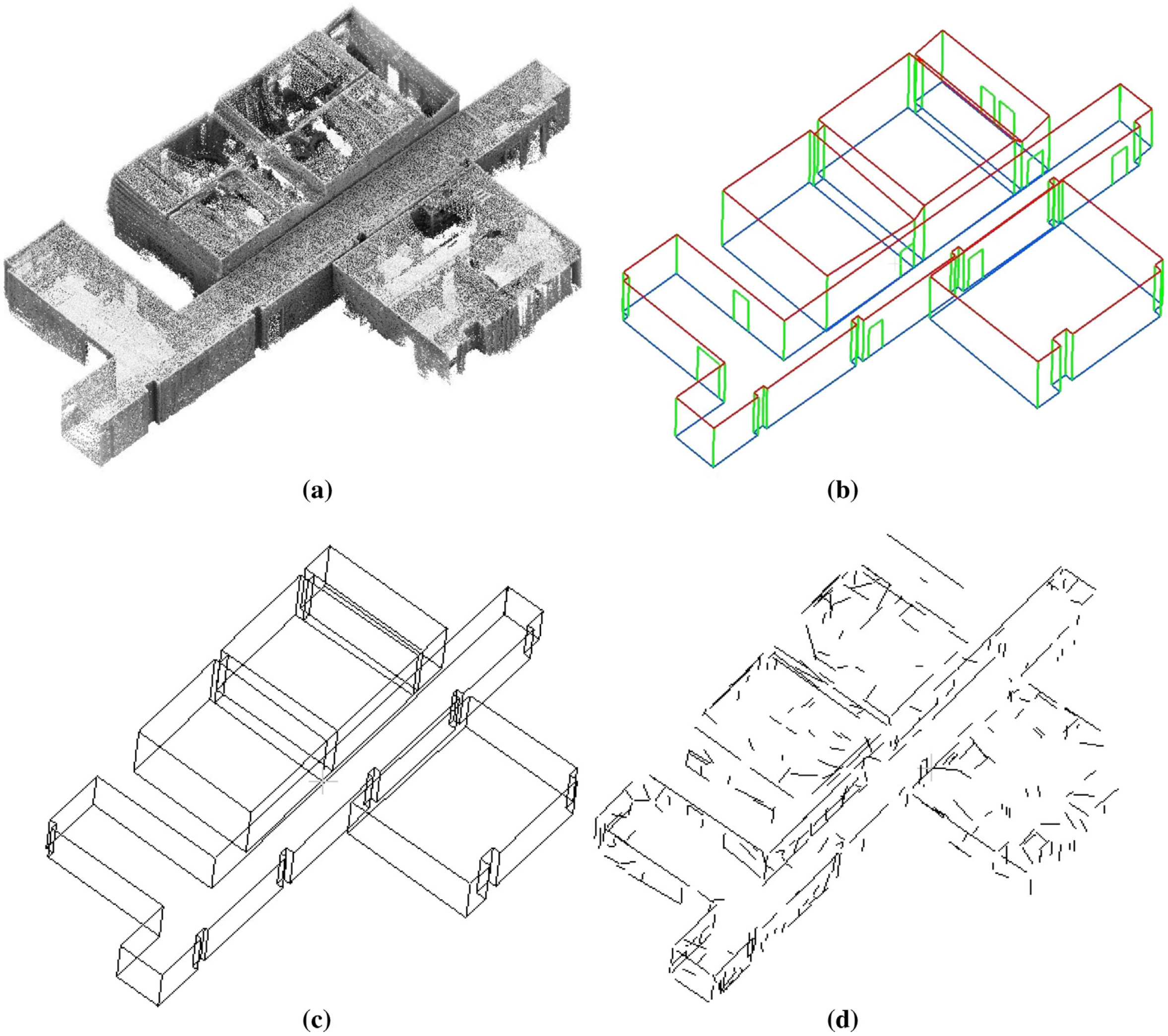

Fig. 19. Result comparison on Scene 4. (a) Raw point clouds. (b) Our method. (c) Oesau's method. (d) Lin's method.

extract the 3D lines directly from 3D point clouds. However, it is impossible to use excessively noisy and detailed lines as inputs for modeling. Instead, we extract lines from different labeled categories and combine the lines extracted from each category. The results show that our process is more suitable for indoor scene point cloud data.

Our method vs. Oesau's method. Following the steps presented in Oesau's method, we tested that method on our data. As shown in Figs. 18 and 19, Oesau's method provides line results, but the wall openings are missing (Figs. 18(c) and 19(c)). Moreover, Oesau's method does not provide the line results for the wall openings, like windows and doors. Also, assuming vertical walls and

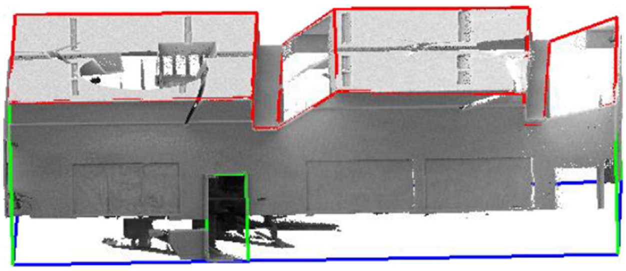

Fig. 21. Line framework extraction result using incomplete terrestrial laser scanner data.

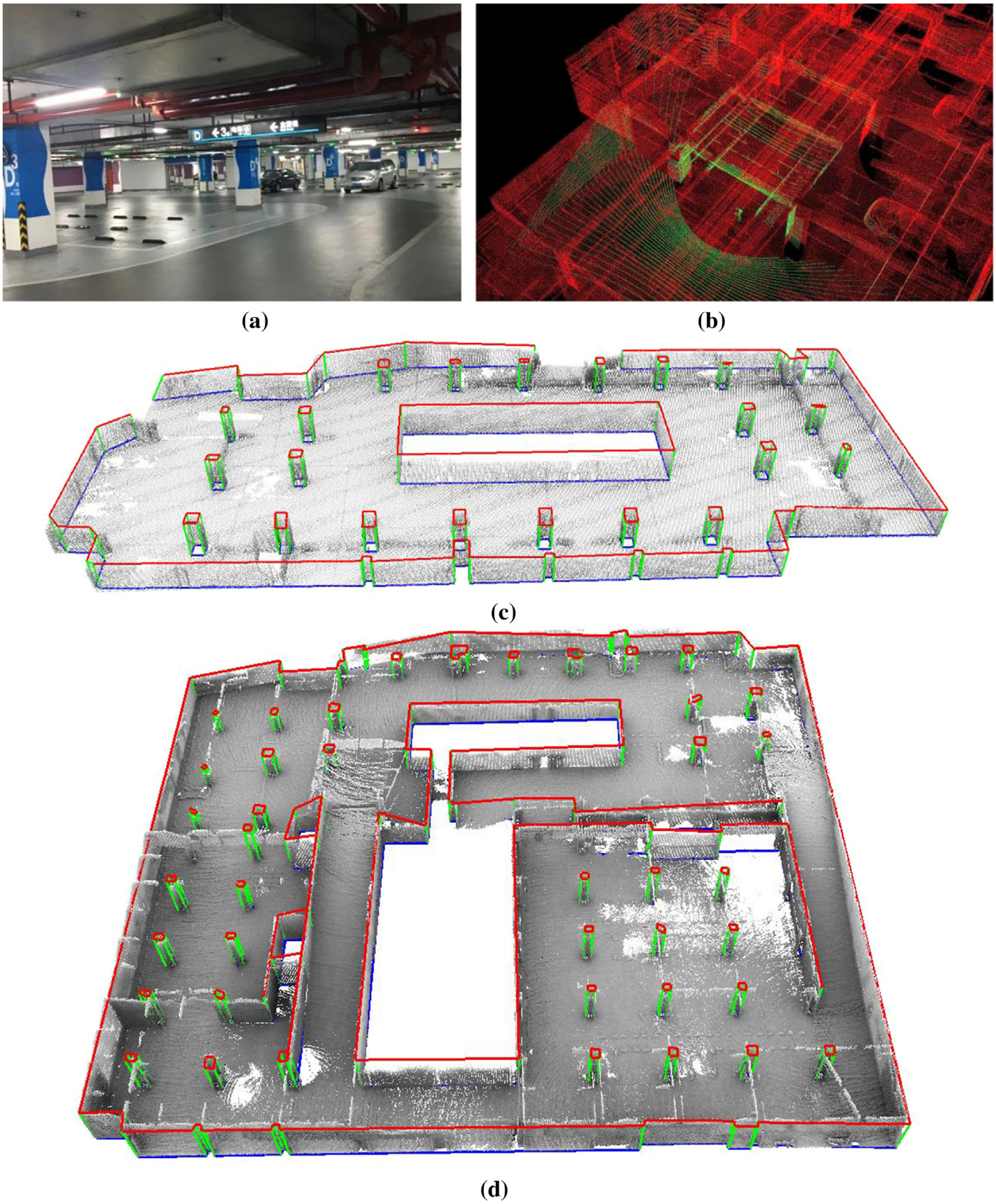

Fig. 20. Line framework extraction results. (a) A photo of Scene 1. (b) Point cloud frame of ceiling in Scene 1. (c) Scene 1. (d) Scene 2.

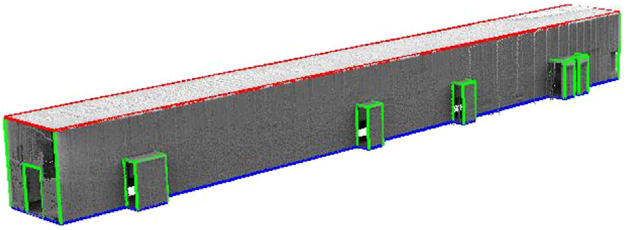

Fig. 22. Part of 11-12-17c data result from http://www.navvis.lmt.ei.tum.de/dataset/.

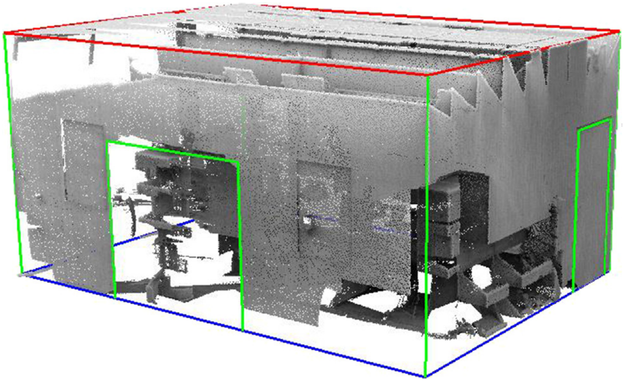

Fig. 23. Part of 2C03 data result from http://www.ifi.uzh.ch/en/vmml/research/datasets.html.

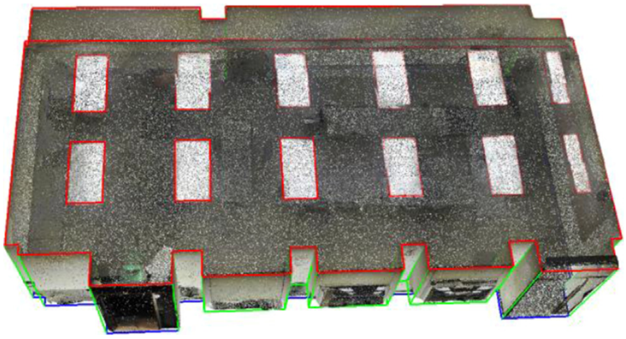

Fig. 24. Conference room data from Stanford 2D-3D-Semantics Dataset (Armeni et al., 2016).

horizontal floors and ceilings, Oesau's method does not provide correct building models when dealing with buildings of inconsistent ceiling height.

Our method provides both optimized structure lines as well as window and door lines (Figs. 18(b) and 19(b)). In addition, because all lines are extracted directly from the 3D point cloud, our method works on buildings with uneven ceiling height. As shown in Figs. 18 (b) and 19(b), two uneven ceiling areas, locating in different planes, are correctly extracted. More importantly, our method provides lines with semantic information, which exactly satisfies the LOD 3 building modeling requirement. In the results, the floor, wall, and ceiling areas are represented by blue, green, and red lines, respectively.

Fig. 20 shows more line framework extraction results using our method on larger testing scenes. Here, both testing scenes are underground parking areas. Unlike common building interiors, the ceilings of these two parking areas are very uneven. As shown in Fig. 20(a) and (b), the background of the ceiling area is cluttered, as noted by the many pipes and parking instruction lights. Our method is effective for obtaining the line framework of these challenging scenes (Fig. 20(c) and (d)). However, due to the incomplete data caused by severe occlusion, some parts of the line are still missing. Also, as shown in Fig. 20(d), the sloping path on the right side is extracted correctly.

In addition, we also tested our method on point cloud data acquired by other systems. For the terrestrial laser scanning RIEGL VZ 1000 data (see Fig. 21), the acquired data is incomplete. Because serious occlusion occurs in this building, most of the floor points are missing. Our method, though, still provides reasonable line framework extraction results with our labeling, 3D line extraction and framework optimization procedures. Moreover, our proposed method extracts the 3D line frameworks from the following four scenes: 2D laser scanner-based corridor (Fig. 22), 3D laser scanner-based single room (Fig. 23), 3D camera-based conference room (Fig. 24) and 2D laser scanner-based multi-floor building (Fig. 25). Especially, the sloping ceiling in Fig. 25 is extracted successfully, indicating that our proposed method is effective for uneven ceiling scenes. The results indicate that out method works well with different types of indoor point cloud data.

7. Conclusion

This paper presented a novel semantic line framework modeling method for indoor environments using backpacked laser scanning point cloud data. We first proposed a patch-based labeling result provided by the CRFs-based method to process 3D modeling

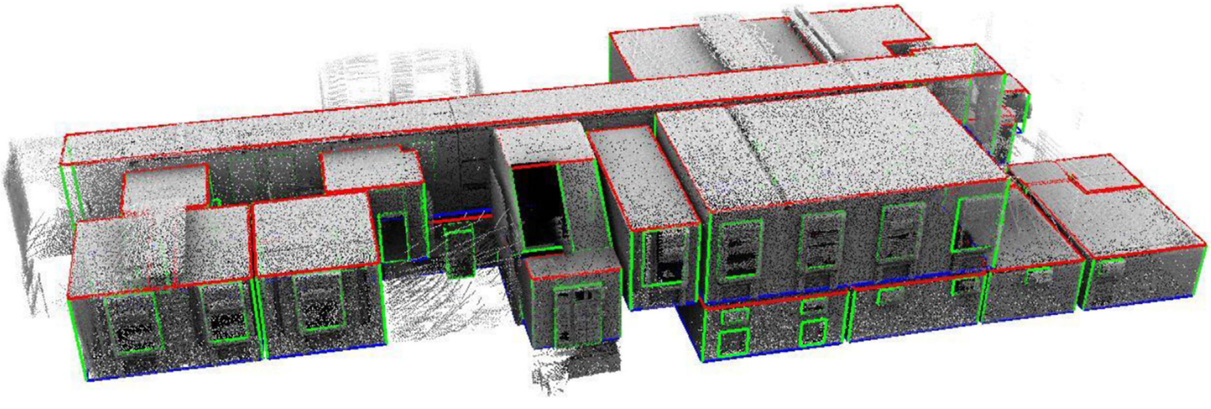

Fig. 25. TUB2 dataset of the ISPRS benchmark on indoor modeling (Khoshelham et al., 2017).

without prior knowledge of the building. Then, to provide a rich structural representation of the cluttered and occluded building interior, we directly extracted a 3D line structure from the 3D point cloud. In addition, we proposed a cGAN-based deep learning model to optimize the line framework. By using different training data, the optimized framework can be further applied to buildings with other structures. Line optimization can further instruct the point cloud quality evaluation to form a closed loop of the data collection and the representation of the indoor environment. Experimental results show that the algorithm provides a good line framework-based semantic model for the indoor point clouds acquired by our self-built backpacked laser scanning system, as well as other laser scanning point clouds.

Acknowledgments

This work was supported by the National Science Foundation of China (Grant No. 61771413 and U1605254) and the Fundamental Research Funds for the Central Universities (Grant No. 20720170047).

References

Armeni, I., Sener O., Zamir A., Jiang H., Brilakis, I., Fisher, M., Savarese, S., 2016. 3D semantic parsing of large-Scale indoor spaces. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1534–1543.

Babacan, K., Jung, J., Wichmann, A., Jahromi, B. A., Shahbazi, M., Sohn, G., Kada, M., 2016. Towards object driven floor plan extraction from laser point cloud. Int. Arch. Photogram., Rem. Sens. Spatial Inform. Sci., 41.

Biljecki, F., Ledoux, H., Stoter, J., Zhao, J., 2014. Formalisation of the level of detail in 3D city modelling. Comput. Environ. Urban Syst. 48, 1–15.

Bosse, M., Zlot, R., Flick, P., 2012. Zebedee: design of a spring-mounter 3D range sensor with application to mobile mapping. IEEE Trans. Robotics 28 (5), 1104–1119.

Boykov, Y., Veksler, O., Zabih, R., 2001. Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. 23 (11), 1222–1239.

Chen, J., Chen, B., 2008. Architectural modeling from sparsely scanned range data. Int. J. Comput. Vision 78 (2), 223–236.

Desolneux, A., Moisan, L., Morel, J.M., 2000. Meaningful alignments. Int. J. Comput. Vision 40 (1), 7–23.

Gong, Z., Wen, C., Wang, C., Li, J., 2018. A target-free automatic self-calibration approach for multibeam laser scanners. IEEE Trans. Instrum. Meas. 67 (1), 238–240.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al., 2014. Generative adversarial networks. In: Advances in Neural Information Processing Systems, pp. 2672–2680.

Ikehata, S., Yang, H., Furukawa, Y., 2015. Structured indoor modeling. In: Proceedings of the International Conference on Computer Vision, pp. 1323–1331.

Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A., 2017. Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5967–5976.

Jain, A., Kurz, C., Thormählen, T., Seidel, H.P., 2010. Exploiting global connectivity constraints for reconstruction of 3D line segments from images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1586–1593.

Jung, J., Hong, S., Yoon, S., Kim, J., Heo, J., 2015. Automated 3D wireframe modeling of indoor structures from point clouds using constrained least-squares adjustment for as-built BIM. J. Comput. Civil Eng. 30 (4), 04015074.

Khoshelham, K., Díaz Vilariño, L., Peter, M., Kang, Z., Acharya, D., 2017. The ISPRS benchmark on indoor modelling. Int. Arch. Photogramm., Rem. Sens. Spatial Inform. Sci., XLII-2/W7, pp. 367–372.

Liu, C., Schwing, A.G., Kundu, K., Urtasun, R., Fidler, S., 2015. Rent3d: Floor-plan priors for monocular layout estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3413–3421.

Lin, Y., Wang, C., Chen, B., Zai, D., Li, J., 2017. Facet segmentation-based line segment extraction for large-scale point clouds. IEEE Trans. Geosci. Rem. Sens. 55 (9), 4839–4854.

Lin, Y., Wang, C., Cheng, J., Chen, B., Jia, F., Chen, Z., Li, J., 2015. Line segment extraction for large scale unorganized point clouds. ISPRS J. Photogramm. Rem. Sens. 102, 172–183.

Luo, H., Wang, C., Wen, C., Cai, Z., Chen, Z., Wang, H., Li, J., 2016. Patch-based semantic labeling of road scene using colorized mobile LiDAR point clouds. IEEE Trans. Intell. Transp. Syst. 17 (5), 1286–1297.

Michailidis, G.T., Pajarola, R., 2016. Bayesian graph-cut optimization for wall surfaces reconstruction in indoor environments. Visual Comput., 1347–1355

Moghadam, P., Bosse, M., Zlot, R., 2013. Line-based extrinsic calibration of range and image sensors. In: Proceedings of the IEEE Conference on Robotics and Automation, pp. 3685–3691.

Munoz, D., Bagnell, J.A., Vandapel, N., Hebert, M., 2009a. Contextual classification with functional max-margin Markov networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 975–982.

Munoz, D., Vandapel, N., Hebert, M., 2009b. Onboard contextual classification of 3-D point clouds with learned high-order Markov random fields. In: Proceedings of the IEEE Conference on Robotics and Automation, pp. 2009–2016.

Mura, C., Mattausch, O., Villanueva, A.J., Gobbetti, E., Pajarola, R., 2014. Automatic room detection and reconstruction in cluttered indoor environments with complex room layouts. Comput. Graphics 44, 20–32.

Ochmann, S., Vock, R., Wessel, R., Klein, R., 2016. Automatic reconstruction of parametric building models from indoor point clouds. Comput. Graphics 54, 94–103.

Ochmann, S., Vock, R., Wessel, R., Tamke, M., Klein, R., 2014. Automatic generation of structural building descriptions from 3d point cloud scans. In: Proceedings of the IEEE Conference on Computer Graphics Theory and Applications, pp. 1–8.

Oesau, S., Lafarge, F., Alliez, P., 2014. Indoor scene reconstruction using feature sensitive primitive extraction and graph-cut. ISPRS J. Photogramm. Rem. Sens. 90, 68–82.

Perez-Yus, A., Lopez-Nicolas, G., Guerrero, J.J., 2016. Peripheral expansion of depth information via layout estimation with fisheye camera. In: Proceedings of the European Conference on Computer Vision, pp. 396–412.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. In: Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 234–241.

Rusu, R.B., Blodow, N., Beetz, M., 2009. Fast point feature histograms (FPFH) for 3D registration. In: Proceedings of the IEEE Conference on Robotics and Automation, pp. 3212–3217.

Sanchez, V., Zakhor, A., 2012. Planar 3D modeling of building interiors from point cloud data. In: Proceedings of the IEEE Conference on Image Processing, pp. 1777–1780.

Schnabel, R., Wahl, R., Klein, R., 2007. Efficient RANSAC for point-cloud shape detection. Comput. Graphics Forum 26 (2), 214–226.

Shapovalov, R., Velizhev, E., Barinova, O., 2010. Nonassociative markov networks for 3d point cloud classification. Int. Arch. Photogramm. Rem. Sens. Spatial Inform. Sci., XXXVIII, Part 3A.

Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. In: Proceedings of the International Conference on Learning Representations.

Taskar, B., Guestrin, C., Koller, D., 2003. Max-margin Markov networks. In: Advances in Neural Information Processing Systems, vol. 16, pp. 25–32.

Taskar, B., Chatalbashev, V., Koller, D., 2004. Learning associative Markov networks. In: Proceedings of the International Conference on Machine Learning, pp. 102–109.

Von Gioi, R.G., Jakubowicz, J., Morel, J.M., Randall, G., 2010. LSD: a fast line segment detector with a false detection control. IEEE Trans. Pattern Anal. Mach. Intell. 32 (4), 722–732.

Wang, W., Sakurada, K., Kawaguchi, N., 2016. Incremental and enhanced scanline-based segmentation method for surface reconstruction of sparse LiDAR data. Rem. Sens. 8 (11), 967.

Wen, C., Pan, S., Wang, C., Li, J., 2016. An indoor backpack system for 2-D and 3-D mapping of building interiors. IEEE Geosci. Rem. Sens. Lett. 13 (7), 992–996.

Wen, C., Qin, L., Zhu, Q., Wang, C., Li, J.J., 2014. Three-dimensional indoor mobile mapping with fusion of two-dimensional laser scanner and RGB-D camera data. IEEE Geosci. Rem. Sens. Lett. 11 (4), 843–847.

Xiong, X., Adan, A., Akinci, B., Huber, D., 2013. Automatic creation of semantically rich 3D building models from laser scanner data. Autom. Constr. 31, 325–337.

Zhang, J., Singh, S., 2014. LOAM: Lidar odometry and mapping in real-time, Robotics: Science and Systems.