ISPRS Journal of Photogrammetry and Remote Sensing

Simple loop closing for continuous LIDAR&IMU Planar Graph SLAM for 3D indoor environments

--Manuscript Draft--

Manuscript Number:	PHOTO-D-21-00539
Article Type:	Research Paper
Section/Category:	Laser scanning
Keywords:	Graph SLAM; loop closure; planar features; point clouds; mobile mapping; LIDAR
Abstract:	Simultaneous localization and mapping (SLAM) is the essential technique in mapping environments that are denied to the global navigation satellite systems (GNSSs), such as indoor spaces. In this article, we present a loop-closing continuous-time LIDAR-IMU SLAM for indoor environments. The design of the proposed SLAM is based on arbitrarily-oriented planar features that allow for point to plane matching for local but also global optimization. Moreover, to update the SLAM graph during the optimization, we propose a simple yet elegant loop closure method in the form of merging the planes together. Representing the SLAM map by planes is advantageous due to the abundant existence of planar structures in indoor built environments. The proposed method was validated on a specific configuration of three 2D LIDAR scanners mounted on a wearable platform (backpack). Scanned point clouds were compared against ones obtained from a commercial mobile mapping system (Viametris iMS3D) and a terrestrial laser scanner (RIEGL VZ-400). The experimental results demonstrate that our SLAM system is capable of mapping multi-storey buildings, staircases, cluttered areas and areas with curved walls. Furthermore, our SLAM system offers comparable performance to that of the commercial system as shown by the low deviation between the point clouds generated by both systems. The majority of the cloud-to-cloud absolute distances – about 92 % – are less than 3 cm.

Simple loop closing for continuous LIDAR&IMU Planar Graph SLAM for 3D indoor environments

S. Karam [1,*], V. Lehtola[1] and G. Vosselman[1]

[1] Dept. of Earth Observation Science, Faculty ITC, University of Twente, 7514 AE Enschede, The Netherlands; (s.karam, v.v.lehtola , george.vosselman)@utwente.nl

* Correspondence: s.karam@utwente.nl; Tel.: +31--53-4894577

Abstract: Simultaneous localization and mapping (SLAM) is the essential technique in mapping environments that are denied to the global navigation satellite systems (GNSSs), such as indoor spaces. In this article, we present a loop-closing continuous-time LIDAR-IMU SLAM for indoor environments. The design of the proposed SLAM is based on arbitrarily-oriented planar features that allow for point to plane matching for local but also global optimization. Moreover, to update the SLAM graph during the optimization, we propose a simple yet elegant loop closure method in the form of merging the planes together. Representing the SLAM map by planes is advantageous due to the abundant existence of planar structures in indoor built environments. The proposed method was validated on a specific configuration of three 2D LIDAR scanners mounted on a wearable platform (backpack). Scanned point clouds were compared against ones obtained from a commercial mobile mapping system (Viametris iMS3D) and a terrestrial laser scanner (RIEGL VZ-400). The experimental results demonstrate that our SLAM system is capable of mapping multi-storey buildings, staircases, cluttered areas and areas with curved walls. Furthermore, our SLAM system offers comparable performance to that of the commercial system as shown by the low deviation between the point clouds generated by both systems. The majority of the cloud-to-cloud absolute distances – about 92 % – are less than 3 cm.

Keywords: Graph SLAM, loop closure, planar features, indoor, point clouds, mobile mapping, laser scanner, LIDAR, planar structures hypothesis, IMU, splines, 6DOF

1. Introduction

Three-dimensional (3D) digital models of indoor environments can be advantageous to professionals from various disciplines such as engineering, architecture and archaeology. Indoor mobile mapping systems (IMMSs) digitize indoor environments quickly and at high levels of detail [1,2]. Such systems have advantages over terrestrial laser scanners (TLS) in terms of time-consumption and labour. The IMMSs rely on simultaneous localization and mapping (SLAM) algorithm for positioning in spaces and environments inaccessible to the global navigation satellite systems (GNSSs).

The core idea of SLAM [3] is to map unknown environments. In this task, problems arise from that these environments may contain pathological geometries and from that the platform motion may be erratic [4] Specifically, light detection and ranging (LIDAR) SLAM degenerates in those pose configurations where the geometry of LIDAR observations is insufficient to estimate the 3D pose of the mapping system. In LIDAR-inertial measurement unit (IMU) SLAM [5,6], the LIDAR provides accurate geometrical data, while the IMU prevents the SLAM from degenerating, for example, when the platform experiences fast rotations. Although the IMU sensor provides good short-term motion estimates [7,8], it suffers from accumulated errors over time due to the dead reckoning-based positioning, and therefore obviously is undesirable as a stand-alone sensor for positioning purposes.

SLAM can be run online to estimate the current pose of the mapping system or offline to optimize the entire trajectory. In the first, only a part of the historic data is used to keep the computational load tractable for real time applications, see e.g. Kalman filter-based SLAMs [9,10]. This is in contrast to the offline SLAM that optimizes the trajectory over the entire recorded data such as graph SLAM [11].

Therefore, the offline SLAM is usually more accurate than the online SLAM. Accordingly, it is often used in mobile mapping where the geometric accuracy of the map is the most important factor.

Typically, graph SLAM uses a discrete pose representation, i.e. it consists of nodes, representing the mapping system poses, connected by edges representing the sensors' measurements that constrain the connected poses [11]. For more accurate and efficient multi-sensor SLAM, continuous-time trajectory representation is a necessity [6,12]. Therefore, our graph SLAM uses splines to represent the trajectory as a continuous six degrees of freedom (6DoF) format [13] so that each LIDAR and IMU measurement have their own timestamp.

Loop closure is one of the key challenges in SLAM because recognizing already visited places requires searching through all the captured data, which becomes computationally intractable as the size of the environment grows. Descriptors [14–17] can be used to reduce the size of to-be-matched information, but the number of stored descriptors, i.e. bag of words, still grows as more data is captured. The bigger the bag, the more complex descriptors are needed to keep them distinguishable from each other. The opposite approach is to use simple descriptors (such as plane parameters) on features that are *large and spatially distinct*. In this work, we rely on such, i.e. planar, features.

In this paper, we propose a loop-closing SLAM system that is capable of producing plane-based maps of various indoor environments in the real world. Our work is built on [6,8,13,18]. The proposed SLAM system has the following six state-of-the-art properties.

1) The proposed SLAM method performs loop closure detection and correction using planar feature-based matching and merging, which is a new contribution (described in Section 3.8).
2) We propose a novel way to reduce the degrees of freedom in the graph optimization problem via a plane parametrization based on a three-fold classification: horizontal, vertical, and slanted rotation (Section 3.2).
3) The trajectory is represented with splines forming a continuous-time model that allows for individual time stamping of each LIDAR point [13] without a need to do IMU pre-integration [5,19] or upsampling to approximate continuous time [12,20].
4) Our SLAM system exploits the IMU to predict the pose of a few successive scans (Section 3.1), similar to e.g. [12,20]. Such a scan-combination allows for our data association technique to segment arbitrarily oriented planar shapes and to allocate them into the SLAM map (Section 3.2). Note that, the term *scan* in this paper refers to a set of three scan lines captured simultaneously by three 2D scanners to form a quasi-3D LIDAR point subset.
5) The local SLAM is 3D and fuses LIDAR and IMU observations [6], briefly introduced in Sections 3.1 and 3.6.
6) Autocalibration of LIDARs is done exploiting the planar feature geometry as in our previous work [18], lately adopted also in [12].

The chosen map representation [13,18] allows for outputting the scanned environments in planar shapes, which is a popular format for the state-of-the-art in indoor 3D reconstruction [21]. This kind of representation is also advantageous because it makes storage easier and data association simpler when compared against a point-based representation, see e.g. [3,22]. We describe experiments in various indoor environments and evaluate the performance of our SLAM system by comparing the obtained point clouds against those obtained from a commercial MMS (Viametris iMS3D) and a TLS (RIEGL VZ-400) used as ground truth.

The data used in our case study is scanned with a specific configuration of three single-layer LIDAR scanners (Hokuyo) that have a limited field of view –270°– i.e. the same one used in the original LOAM method [23]. The scanners are mounted on a wearable platform (backpack).

The remainder of this paper is structured as follows: In the following section, we present the related works. In Section 3, we introduce the overall concept of the proposed SLAM system including planar segments extraction, parametrization, data association and the loop closure technique. Section 4

presents the mobile mapping system used for data collection, the study areas, and the experimental results. Finally, the paper draws conclusions in Section 5.

2. Related Works

The closest works to ours in respect of planar SLAM are IN2LAAMA [12], LIPS [5] and planar bundle adjustment (PBA) [19]. All these works utilize planar segments extracted from LIDAR data using slightly different segmentation methods. The least squares problem in LIPS seeks to minimize the plane-to-plane residuals. In contrast, our SLAM, IN2LAAMA and PBA utilize point-to-plane residuals. Compared to the plane-to-plane based minimization, experiments in [19] show that the point-to-plane based minimization leads to more accurate results. In contrast to LIPS and PBA, we have, similar to IN2LAAMA, a continuous-time trajectory representation, which is necessary to output high-accuracy maps. However, IN2LAAMA [12] upsamples IMU preintegration, which uses a lot more memory compared to our continuous spline formulation of the trajectory, while still containing a discretization residual. Also, IN2LAAMA performs a loop closure detection based on iterative closest point (ICP)-like pose proximity of individual LIDAR observations, which leads to redundant loop closures without a given time threshold, and is similar to what we use for local SLAM (Section 3.6). In contrast, our global loop closure detection analyzes the relationship between the large and spatially distinct planes in the global SLAM map (Section 3.8). This loop closure technique is one of the main contributions in this work. Another contribution is the reduction in the degrees of freedom of the planes based on the LIDAR observations.

Other works include 2D methods, e.g. [22], methods assuming vertical walls, e.g. [24] and a planar feature-based SLAM system experimenting with dividing and projecting the 3D LIDAR data onto three image planes before applying planar segmentation [25].

Apart from graph slam methods, local planarity of points is exploited also in real-time SLAM. The formulation in [26] to describe point planarity is sufficiently fast to be used with Kalman filter-based estimators and has proven to be successful in [10,23,27]. In contrast to being indoors, planar descriptors for points are not necessarily needed but do improve the quality of the results also outdoors when planar zones are large and spatially distinct such as building façades or walls [28].

3. Methodology

The proposed SLAM system is a planar feature-based SLAM algorithm that is designed to map building interiors with arbitrarily oriented planes and inherently performs loop closure. The overall concept, with its key components, is shown in Figure 1. After initialization (Section 3.1) we extract planar segments from LIDAR data (Section 3.2), assign these segments to planar features (Section 3.2), present the trajectory optimization problem (Sections 3.3-3.7), and our loop closure technique (Section 3.8). The final outputs of the introduced SLAM system are reconstructed 3D planes, a 3D point cloud and the 3D trajectory that the mapping system followed while scanning (Figure 6). The generated point cloud, together with trajectory information, can be used for the semantic interpretation [29] or space partitioning [30]. Moreover, the normal vectors of the reconstructed planes point towards the system's trajectory, which makes our SLAM beneficial for 3D reconstruction.

3.1. Initialization

The model coordinate system ($\boldsymbol{m}$) (or the world coordinate system) in which the final indoor model will be described is established based on a few independent planes extracted from the first few LIDAR scans, as explained in [13]. These planes form the initial state of the Local map and the start point of the platform trajectory is defined as the origin of the model system. The Local-SLAM phase starts by predicting the pose of a few successive scans in the model system using the IMU data, thereby forming what we term **scan-combination** as a collection of the predicted successive scans. Here, a scan-

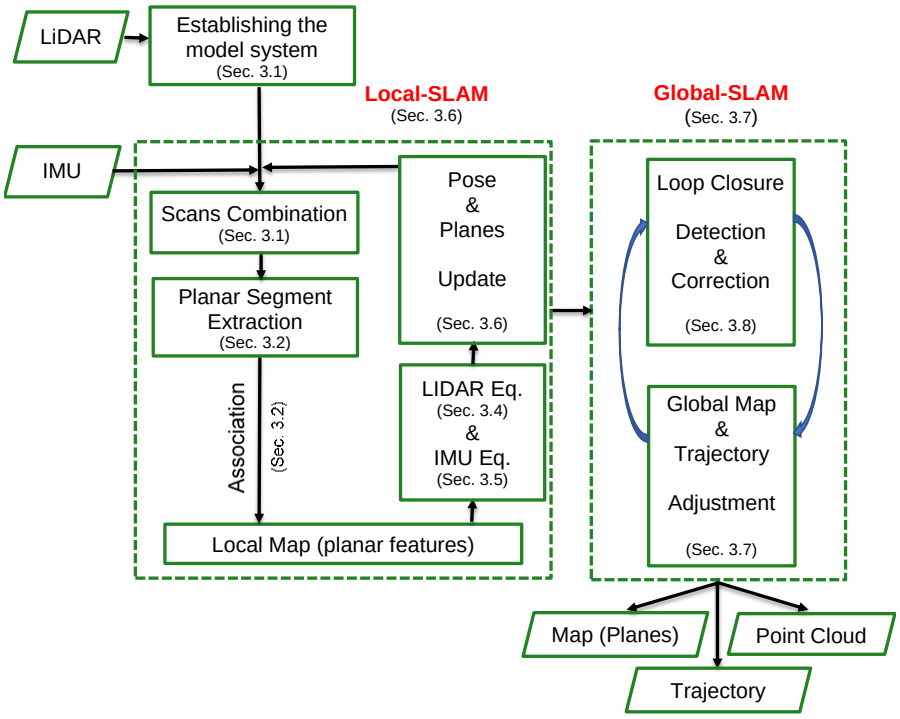


Figure 1. The overall concept of the loop-closing LIDAR-IMU SLAM system

combination consists of 10 scan lines captured during 0.25 sec. Such scan-combination is largely hardware-independent, i.e. it can be formed from almost any type of LIDAR and inertial combination. Similar approach is used also in [12,20].

3.2. Planar Segment Extraction and local SLAM map updating

To extract planes from the scan-combination (defined in Section 3.1), we run a surface growing segmentation [31] and fit a plane to the points of each segment. Figure 2 shows the extracted planes from a scan-combination captured on a staircase. A 2D oriented bounding box that represents the extent of the plane is extracted from its points.

In order to limit the amount of free plane parameters, we classify the fitted planes to the planar segments into horizontal, vertical and slanted planes, described by respectively 1, 2, and 3 parameters. Planes are instantiated as slanted when insufficient data is available (relatively narrow segments) to reliably classify them as horizontal or vertical. As the SLAM progresses, more scan line points are assigned to the same plane (as described below). This then may allow to better understand the plane's orientation. If that is the case the parameterization of the plane is updated accordingly.

Furthermore, to increase the robustness of the hypothesis generation, we filter out those planes that have a standard deviation of plane fitting residuals σ_{p-Pl} higher than max_σ_{p-Pl} or a number of points N_p lower than min_N_p. Moreover, only planes that were extracted from more than one scan are included in the estimation process.

This plane extraction allows for the LIDAR data to be robustly matched against the up-to-date version of the local map in SLAM. Specifically, our data association technique seeks to find correspondences between two groups of planes: (i) these planar segments (plane hypotheses) extracted recently at t_r ($r = \mathrm{n}+1, \ldots, \mathrm{N}$) from the scan-combination captured in $N - \mathrm{n} = 10$ time steps, i.e. 0.25 sec; and (ii) SLAM map planes (plane features) estimated in the previous time steps $t_p, p = 1,2, \ldots, n$.

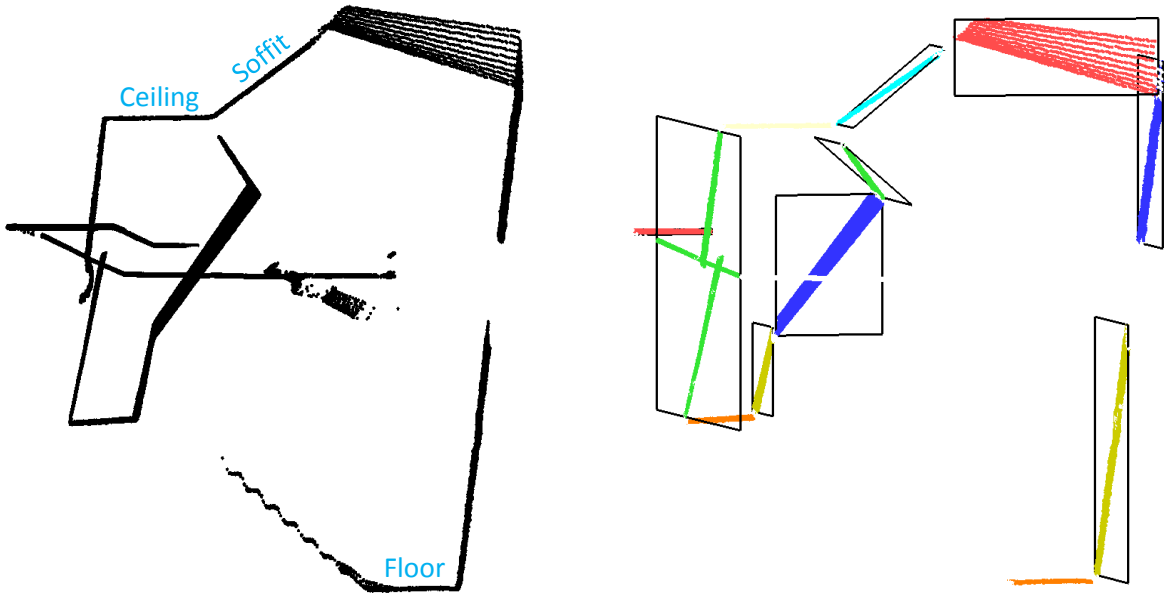


Figure 2. Planar segment extraction. Left: a scan-combination consists of a few successive scans (10) captured on a staircase. Right: surface growing segmentation with the fitted planes shown as 2D bounding boxes. For visualisation purposes, the planes for ceiling and floor's segments are not shown.

A hypothesis and a feature may represent the same planar structure; thus, they will be matched if their 2D bounding boxes overlap and if the distance and angle between them are within a certain limit. Consequently, the planar segment's points are added to the matched plane in the Local-SLAM map; this requires an update of its 2D bounding box if the added points are outside the already seen part of that plane. The rest of the extracted planes – those that have not been matched with any of the previously estimated planes – are added to the SLAM map as new planes.

However, closely aligned scan lines not spanning a large area do not allow for a robust estimation of a normal vector of a plane. Therefore, they are not used to generate a new plane hypothesis. The points in such scan lines are, however, still used if they can be associated to an already existing plane feature. Additionally, each point that was not part of a planar segment (i.e. is still un-associated) is associated to its closest plane feature in the map, if that point is located within the 2D bounding box of that plane and within a certain distance.

After these steps, the SLAM map is updated. Some planes may have been extended and new planes may have been instantiated. As a consequence, 2D bounding boxes of nearby planes with similar normal may now overlap. Therefore, we perform a further association check between all planes in the map. The association check is similar to the plane-to-plane association step above with the only difference that here we always merge the plane with fewer points into the plane with a larger number of points as that is considered to be more reliable. This association helps to avoid multiple instantiations of the same planar structure and decreases the number of planes, which in turn reduces the complexity of the SLAM model and speeds up the mapping process.

3.3. State definition and Plane and Trajectory Parametrization

The extracted planes are defined in the model coordinate system by the Hessian plane model:

$$np - d = 0 \tag{1}$$

where $n = (n_x, n_y, n_z)$ is the plane's normal vector, p is a point lying in the plane and d is the signed distance to the plane (from the origin along the normal vector). Hence, plane h can be expressed with angular parameters $(\theta_h, \phi_h, d_h)^T$ with an identical amount of free parameters. Note that, in our SLAM, the normal vector of a plane (n) always points towards the system's trajectory. Additionally,

the area of a plane is represented by a 2D oriented bounding box (Figure 2) which grows as additional observations are added from the scanning, and this bounding plays an important role in the data association as described in Section 3.2.

The platform pose parameters (x, y, z, ω, ϕ, κ) are modelled as continuous functions over time using cubic splines [18] which guarantees a level of smoothness for the final trajectory and allows for a continuous time representation for individual measurements. For instance, roll ω is formulated as follows:

$$\omega(t) = \sum_i \alpha_{\omega,i}.\, B_i(t), \tag{2}$$

where $\alpha_{\omega,i}$ is the spline coefficient for ω to be estimated on interval i and $B_i(t)$ is the B-spline value at time t on interval i.

Our SLAM is an optimization-based method that combines the LIDAR and IMU measurements together to estimate the state of the system and the SLAM map. Instead of defining state vectors for the trajectory, we define them for the observations. Adding together all LIDAR and IMU observations, the total state to be estimated X_{total} consists of spline coefficients for each node interval i, where i indexes the M spline intervals, and parameters of all planes, as follows:

$$X_{total} = [\alpha_1,\ \alpha_2, \dots \dots \alpha_M,\quad P_1,\ P_2, \dots \dots\ P_H\ ] \tag{3}$$

where j refers to the number of LIDAR observation, $j = 1,2, \dots \mathrm{K}$, $\alpha_i = [\alpha_{\omega,i}, \alpha_{\varphi,i}, \alpha_{\kappa,i}, \alpha_{x,i}, \alpha_{y,i}, \alpha_{z,i}]$ spline coefficients on interval i and $P_h = (\theta_h, \phi_h, d_h)$ is the parametrization of plane h, $h = 1,2, \dots, H$.

Here, we include the map features into the state because our purpose is to build accurate maps for mapping purposes using graph SLAM. We first formulate the LIDAR (Section 3.4) and IMU (Section 3.5) observation equations that form the equation system for Local_SLAM (Section 3.6) and Global_SLAM (Section 3.7).

3.4. LIDAR observation equation

The LIDAR observation equation is formulated based on the expectation that the distance between a point p and its assigned plane $P_h = (n_h, d_h)$ equal zero [13,18]. Due to the sensor noise and estimation errors, this distance is not necessarily zero, but a residual r_p.

Consequently, for a point $p_j^m = (x_j^m, y_j^m, z_j^m)$ in the model coordinate system (m), the equation can be formulated as follows:

$$r_{p_j^m} = n_h\, p_j^m - d_h, \tag{4}$$

The normal vector of a plane can be parametrized using the spherical coordinates (azimuth θ, inclination ϕ) and the observation equation can be rewritten as

$$r_{p_j^m} = cos\phi_h\, cos\theta_h\ x_j^m + cos\phi_h\, sin\theta_h\, y_j^m + sin\phi_h\ z_j^m - d_h, \tag{5}$$

Accordingly, the plane P_h is represented as $P_h = (\theta_h,\ \phi_h,\ d_h,\ OBB_h)$ where OBB_h contains information about the 2D bounding box extents. The mapping system records the point p_j at time t_j in its platform coordinate system (f), which is later transformed to the model system by:

$$p_j^m = R_f^m(t_j)\ p_j^f + v(t_j), \tag{6}$$

where $R_f^m(t_j)$ and $v(t_j)$ are the time-dependent rotation matrix and translation from the platform system (f) to the model system (m), respectively.

The continuous representation of the trajectory using splines allows the pose parameters to be derived at any time. As each recorded LIDAR point p_j^f has its own timestamp t_j, our SLAM transforms the point p_j^f to the model system with rotation $R_f^m(t_j)$ and translation $v(t_j)$ parameters derived from the pose splines at its timestamp t_j as follows:

$$R_f^m(t_j) = R(\sum_i \alpha_{\omega,i}.B_i(t_j)\ , \sum_i \alpha_{\varphi,i}.B_i(t_j), \sum_i \alpha_{\kappa,i}.B_i(t_j)) \tag{7}$$

$$v(t_j) = (\sum_i \alpha_{x,i}.B_i(t_j)\ , \sum_i \alpha_{y,i}.B_i(t_j), \sum_i \alpha_{z,i}.B_i(t_j)) \tag{8}$$

By substituting Eq. (6) in Eq. (4), we obtain the following observation equation:

$$r_{p_j^m} \stackrel{\text{def}}{=} n_h\left[R_f^m(t_j)\ p_j^f + v(t_j)\right] - d_h, \tag{9}$$

After linearizing Eq. (9) with respect to the unknown pose spline coefficients and unknown plane parameters [13] using Taylor-series expansion, the residual $r^0_{p_j^m}$ for LIDAR observations is

$$\underbrace{{n_h}^0\left[\left(R_f^{m0}(t_j)\ p_j^f + v^0(t_j)\right)\right] - {d_h}^0}_{r^0_{p_j^m}} = \tag{10}$$

$$-{n_h}^0\frac{\partial R_f^{m0}(t_j)}{\partial\omega}p_j^f\sum_i \Delta\alpha_{\omega,i}.B_i(t_j) -$$

$${n_h}^0\frac{\partial R_f^{m0}(t_j)}{\partial\varphi}p_j^f\sum_i \Delta\alpha_{\varphi,i}.B_i(t_j) -$$

$${n_h}^0\frac{\partial R_f^{m0}(t_j)}{\partial\kappa}p_j^f\sum_i \Delta\alpha_{\kappa,i}.B_i(t_j) -$$

$$n_{h_x}^0\sum_i \Delta\alpha_{v_x,i}.B_i(t_j) - n_{h_y}^0\sum_i \Delta\alpha_{v_y,i}.B_i(t_j) - n_{h_z}^0\sum_i \Delta\alpha_{v_z,i}.B_i(t_j) +$$

$$\left[sin\,{\theta_h}^0 cos{\phi_h}^0\ \ -cos{\theta_h}^0 cos{\phi_h}^0\ \ 0\right]\left(R_f^{m0}(t_j)p_j^f + v^0(t_j)\right)\Delta\theta_h\ +$$

$$\left[cos\,{\theta_h}^0 sin{\phi_h}^0\ \ sin{\theta_h}^0 sin{\phi_h}^0\ \ -cos{\phi_h}^0\right]\left(R_f^{m0}(t_j)p_j^f + v^0(t_j)\right)\Delta\phi_h\ +\ \Delta d_h$$

where the upper index 0 denotes the approximate value, $\Delta\alpha_{\omega,i}$, $\Delta\alpha_{\varphi,i}$, $\Delta\alpha_{\kappa,i}$, $\Delta\alpha_{v_x,i}$, $\Delta\alpha_{v_y,i}$, $\Delta\alpha_{v_z,i}$ are the unknown increments of the pose splines coefficients on interval i and $\Delta\theta_h, \Delta\phi_h, \Delta d_h$ are the unknown increments of the parameters of plane P_h.

Eq. (10) is written in general form, i.e. for arbitrarily oriented planes for which all three increments ($\Delta\theta$, $\Delta\phi$, Δd) are estimated. However, we reduce the problem by reducing the degrees of freedom of planes, depending on the plane type. For horizontal planes, we set $\theta = 0$, $\phi = 0$. For vertical planes, we set $\phi = 0$. This increases the robustness of the optimization.

3.5. IMU observation equation

The IMU in the proposed SLAM is utilized not only in the pose prediction but also in the pose estimation. From several strategies developed by Karam et al. [6] to integrate the IMU with the LiDAR SLAM, this paper uses the joint estimation strategy because it is the most robust. The equation system in this strategy consists of two types of observation equations: the LiDAR observation equation, Eq.

(10), that is formulated for each point assigned to a plane in the SLAM map; and the IMU observation equation that is formulated for each measurement along one of the axes of the IMU sensor system (*s*). As the IMU measures angular velocity $\dot{V}^s_{imu} = (\dot{\omega}^s_{imu}, \dot{\varphi}^s_{imu}, \dot{\kappa}^s_{imu})$ and acceleration $\ddot{T}^s_{imu} = (\ddot{X}^s_{imu}, \ddot{Y}^s_{imu}, \ddot{Z}^s_{imu})$ along all three axes over time, we formulate six IMU equations at each IMU timestamp t_u, Eq. (14) and Eq. (15). The IMU observation equations have been described in [6], but for completeness, they are also described here with additional details.

The acceleration observation equation is formulated based on the expectation that the IMU acceleration, after the rotation to the model system and the gravity (g) compensation, should correspond to the second-order derivative of the mapping system's location $\ddot{T}^m_f = (\ddot{X}^m_f, \ddot{Y}^m_f, \ddot{Z}^m_f)$.

$$\ddot{T}^m_f(t_u) = R^m_f(t_u)\, R^f_s\, \ddot{T}^s_{imu}(t_u) - \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} \tag{11}$$

where $R^f_s = R_z(90°)$ is the time-independent rotation matrix from the IMU sensor system to the frame system.

However, both sides of Eq. (11) are not necessarily equal and the difference is termed acceleration residual $r_{\ddot{T}}$ in this paper.

$$\ddot{T}^m_f(t_u) - R^m_f(t_u)\, R^f_s\, \ddot{T}^s_{imu}(t_u) + \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} = r_{\ddot{T}} \tag{12}$$

Similarly, the following angular velocity observation equation is formulated based on the expectation that the IMU angular velocity should correspond to the first-order derivatives of the mapping system's rotation angles $\dot{V}^m_f = (\dot{\omega}^m_f, \dot{\varphi}^m_f, \dot{\kappa}^m_f)$ after the rotation to the IMU sensor system.

$$\dot{V}^s_{imu}(t_u) - R^S_f(t_u) S^f_m(t_u) \dot{V}^m_f(t_u) = r_{\dot{V}} \tag{13}$$

where $r_{\dot{V}}$ is the angular velocity residual at t_u, $R^S_f = \left(R^f_s\right)^T$ and $S^f_m = \begin{pmatrix} \cos\varphi\cos\kappa & \sin\kappa & 0 \\ -\cos\varphi\sin\kappa & \cos\kappa & 0 \\ \sin\varphi & 0 & 1 \end{pmatrix}^f_m$ is the transformation matrix from the model system to the frame system; see [6] for more detail.

As the pose parameters in Eq. (10) are modelled using splines, the IMU measurements (acceleration and angular velocity) in the IMU equations are also modelled using splines. Although the IMU has a different (usually higher) sensing frequency than the LIDAR scanner, the use of splines enables SLAM to derive the mapping system's location and orientation at any point of time and forms Eq. (12) and Eq. (13). As splines are polynomial functions, it is straightforward to derive the accelerations ($\ddot{T}^m_f$) as the second-order derivatives of the translation splines (T^m_f) and the angular velocities ($\dot{V}^m_f$) as the first-order derivatives of the rotation splines.

For example, for the translation X spline $X(t) = \sum_i \alpha_{x,i}\, B_i(t)$, the acceleration $\ddot{X}$ spline becomes $\ddot{X}(t) = \sum_i \alpha_{x,i}\, \ddot{B}_i(t)$, where $\alpha_{x,i}$ is the X spline coefficient to be estimated on interval i. For the rotation angle ω spline $\omega(t) = \sum_i \alpha_{\omega,i}\, B_i(t)$, the angular velocity $\dot{\omega}$ spline becomes $\dot{\omega}(t) = \sum_i \alpha_{\omega,i}\, \dot{B}_i(t)$, where $\alpha_{\omega,i}$ is the ω spline coefficient to be estimated on interval i.

Hence, both the pose parameters and the IMU measurements are expressed in terms of the same to-be-determined spline coefficients.

Similarly to Eq. (9), we linearize Eq. (12) and Eq. (13) with respect to the unknown pose spline coefficients. Hence, the linearized acceleration and angular velocity observation equations becomes the following Eq. (14) and Eq. (15), respectively.

$$\underbrace{\ddot{T}_f^{m^0}(t_u) - R_f^{m^0}(t_u) R_s^f \ddot{T}_{imu}^s(t_u) + \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix}}_{r_{\ddot{T}}^0} = -\begin{pmatrix} \sum_i \Delta\alpha_{v_x,i} \ddot{B}_i(t_u) \\ \sum_i \Delta\alpha_{v_y,i} \ddot{B}_i(t_u) \\ \sum_i \Delta\alpha_{v_z,i} \ddot{B}_i(t_u) \end{pmatrix} +$$

$$\frac{\partial R_f^{m^0}(t_u)}{\partial \omega} R_s^f \ddot{T}_{imu}^s(t_u) \sum_i \Delta\alpha_{\omega,i} B_i(t_u) \quad (14)$$

$$+ \frac{\partial R_f^{m^0}(t_u)}{\partial \varphi} R_s^f \ddot{T}_{imu}^s(t_u) \sum_i \Delta\alpha_{\varphi,i} B_i(t_u) + \frac{\partial R_f^{m^0}(t_u)}{\partial k} R_s^f \ddot{T}_{imu}^s(t_u) \sum_i \Delta\alpha_{\kappa,i} B_i(t_u)$$

$$\underbrace{\dot{V}_{imu}^s(t_u) - R_f^s S_m^{f^0}(t_u) \dot{V}_f^{m^0}(t_u)}_{r_{\dot{V}}^0}$$

$$= R_f^s S_m^{f^0}(t_u) \begin{pmatrix} \sum_i \Delta\alpha_{\omega,i} \dot{B}_i(t_u) \\ \sum_i \Delta\alpha_{\varphi,i} \dot{B}_i(t_u) \\ \sum_i \Delta\alpha_{\kappa,i} \dot{B}_i(t_u) \end{pmatrix} \quad (15)$$

$$+ R_f^s \frac{\partial S_m^{f^0}(t_u)}{\partial k} \dot{V}_f^{m^0}(t_u) \sum_i \Delta\alpha_{k,i} B_i(t_u) + R_f^s \frac{\partial S_m^{f^0}(t_u)}{\partial \varphi} \dot{V}_f^{m^0}(t_u) \sum_i \Delta\alpha_{\varphi,i} B_i(t_u)$$

Eq. (14) and Eq. (15) contain only the unknown increments of the pose splines coefficients compared to Eq. (10) which also contains the unknown increments of the plane parameters. The IMU is pre-calibrated, see Section 4.5 for details.

3.6. Local_SLAM

After each data association that associates the extracted segments in the scan-combination to the local map, Eq. (10) is formulated for each point assigned to a plane in the local map. As we commented above, Local-SLAM runs within a local time window that is longer than the time interval of the scan-combination. At each IMU timestamp t_u within this time window, Eq. (14) and Eq. (15) are formulated. Consequently, the equation system in the Local-SLAM consists of Eq. (10), Eq. (14) and Eq. (15) and is solved by a least-squares solution to estimate the unknown increments ΔX containing increments of all pose splines coefficients and plane parameters used within the Local_SLAM time window. The cost function, Eq. (16), that the least squares solution seeks to minimize is the sum of squared point-to-plane (r_p), acceleration ($r_{\ddot{T}}$) and angular velocity ($r_{\dot{V}}$) residuals.

$$\underbrace{argmin}_{X_{total}} \left[\sum_j \left\| r_{p_j}^0 \right\|^2 + \sum_{j_{imu}} \left\| r_{\ddot{T}_{j_{imu}}}^0 \right\|^2 + \sum_{j_{imu}} \left\| r_{\dot{V}_{j_{imu}}}^0 \right\|^2 \right] \quad (16)$$

where $j = 1,2, \dots K$ and $j_{imu} = 1,2, \dots J_{imu}$ where K and J_{imu} refer to the number of LIDAR points and the number of IMU measurements involved in Local_SLAM, respectively.

Using the estimated increments ΔX, the pose and plane parameters X within the Local-SLAM time window are updated, see Eq. (17). The update process iterates and all splines are updated, to withhold the smoothness of the trajectory, until convergence.

$$X_{total,i_t+1} = X_{total,i_t} + \Delta X \tag{17}$$

where i_t is the number of iteration.

At the end of each Local-SLAM, we generate a new scan-combination by predicting the pose parameters using the IMU data and extract the existing planes to be associated with the local map. Thus, the IMU in our SLAM is not only used to generate the graph, but it is also used to build the data association. The pose and planes parameters within the new time window are updated, using Eq. (17), in the local map by invoking again the Local-SLAM as addressed above.

3.7. Global-SLAM and autocalibration

Compared to Local_SLAM, Global-SLAM process includes also the estimation of the intrinsic sensor calibration parameters of all three scanners (auto-calibration) and the registration parameters describing the relative poses of the scanners (auto-registration). These registration and calibration parameters are updated within SLAM process [18]. Consequently, the total state to be estimated X_{total} in Global_SLAM becomes

$$X_{total} = [\alpha_1,\ \alpha_2, \dots\dots \alpha_M,\quad P_1\ P_2 \dots\dots\ P_H\quad \mu\quad \Psi] \tag{18}$$

where μ is 8 x 1 vector contains the intrinsic sensor calibration parameters of all three scanners, except the range scale factor of one scanner, and ψ is 11 x 1 vector contains the parameters describing the relative poses of the scanners, except the vertical offset between one scanner and the other two; see [18] for more detail.

After processing the whole dataset by Local-SLAM, Global-SLAM phase starts by invoking a loop closure technique that works as described in the next section. Next, one integral adjustment solves the equation system that consists of Eq. (10), Eq. (14) and Eq. (15) of all assigned LIDAR points and IMU measurements in the dataset. The least-squares optimization framework jointly optimizes the system poses and parameters of planes by minimizing the above cost function, Eq. (16), overall data. Similar to the Local_SLAM, the estimation process updates the total state X_{total} iteratively (see Eq. (17)) until convergence. As a result, we have estimates of the calibration (μ) and registration (Ψ) parameters, the whole trajectory, i.e. all pose spline parameters, as well as all parameters of m planes in the model coordinate system that form the Global map.

3.8. Loop Closure Detection and Correction

The built map in our SLAM consists of planar features. Therefore, if the system has revisited a place and we have a loop closure situation, there will be two groups of planes reconstructed in this loop closure place. The first group has been reconstructed during the first visit to the place (coloured green in Figure 4) and the second group has been reconstructed during the second visit (coloured blue in Figure 4).

Therefore, in the Global-SLAM phase (see Figure 1), the loop closure detection searches in the map for a pair of planes that meet the following criteria. First, the normal vectors of the plane-pair are pointing towards the same direction. Second, the 2D bounding boxes of the plane-pair overlap. Third, the pair of planes are separated in time, i.e. have been captured at clearly separate time instances.

If such pairs of planes are detected, they are merged to modify the graph to perform loop closure correction. The graph modification is implemented through an integral adjustment process, as described in Section 3.7. The loop closure and update process, which represent the Global-SLAM framework, iterates until convergence, as shown in Figure 1.

Note that merging planes is our way of modifying the SLAM graph to correct misclosure at the end of a loop. As the planar features are large (minimum area condition in Section 3.2) and spatially distinct, the risk of merging wrong planes is almost negligible. The plane parametrization (Section 3.3) and division into classes (Section 3.2) also further prevents the risk of merging wrong planes.

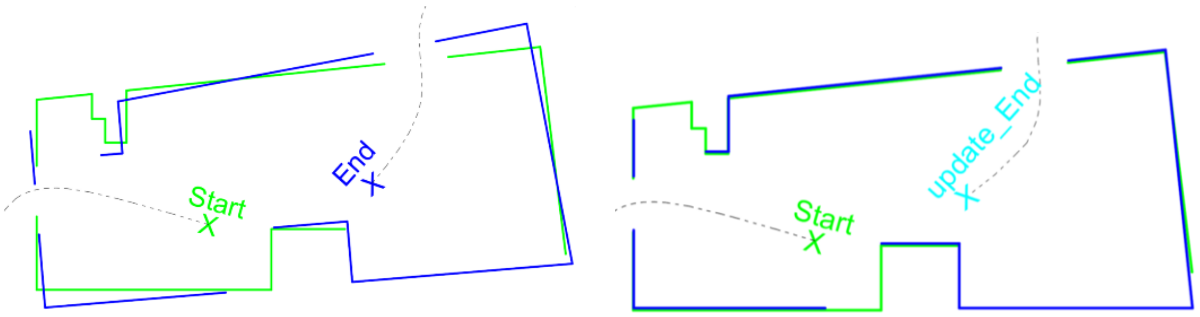


Figure 4. The loop closure schematic shows the detected planes in the loop closure area (room) from a top perspective. Left: The misclosure at the start (green)-end (blue) room. Right: After loop closure correction. For visualization purposes, the detected planes are shown without merging.

4. Experiments

In this section, we first briefly describe the mapping system used for data collection (Section 4.1) and then give an overview of the sites where the datasets were collected (Section 4.2). Next, we present the experimental results which are divided into two parts. First, we present the results of our closing-loop SLAM system in various indoor environments (Section 4.3). Second, we compare the obtained point clouds to ones derived from a commercial MMS (Viametris iMS3D) and a TLS (RIEGL VZ-400) (Section 4.4).

4.1. Mobile Mapping System

Figure 5 shows the equipment used for the data collection. It is our research multi-sensor backpack MMS (ITC-Backpack) in which the core is formed by the triple-Hokuyo LiDAR scanners (laser range finders) [18]. The system configuration consists of one top scanner mounted horizontally and on a level that prevents occlusion problem by the operator, and the other two scanners are tilted and mounted to the left and right of the top scanner. Furthermore, an Xsens IMU is positioned horizontally underneath the top scanner. The selected scanners' configuration provides a quasi-3D LiDAR point subset that consists of three scan lines at each timestamp.

4.2. Study Areas and Datasets

We used two study areas that differ in terms of geometry, architecture and cluttering in general. The first study area is the building of ITC faculty at the University of Twente, The Netherlands. That building is a non-Manhattan world-building and has an unusually shaped design featuring slanted planes. The second study area is the Fire Brigade building in Haaksbergen, The Netherlands. That is composed of two floors and has many small rooms (dressing, laundry, showers, toilets, storage spaces) and many rooms have several doors. The building's architecture features rounded walls and a circular bar.

To verify the ability of the loop-closing SLAM system, experiments were carried out in seven indoor application scenarios within these two study areas. That generated seven datasets, as shown in Table 1. The data collection duration for each dataset is listed in the third column. The fourth column lists the key characteristics that were the reasons for capturing each dataset. In other words, the listed characteristics sometimes form obstacles making the captured datasets challenging and suitable to rigorously test the ability of our SLAM system.

We have selected the listed scenarios in a way that the proposed SLAM can be tested in: a multi-storey space and an environment with arbitrarily oriented planes (ITC_f2f3f2_Loop and ITC_f2f5_StairCase), various staircases (ITC_f2f5_StairCase and FB_f0f1_Stairs), long and narrow corridors (ITC_f2f3f2_Loop and ITC_f1_LargeLoop) and a cluttered area with curved surfaces (FB_f1).

Figure 5. Mobile mapping system (ITC Backpack) used for the data collection. The Velodyne scanner observations have not been used in this work. Left: scanning the central staircase in the ITC building. Right: scanning the dressing room in the Fire Brigade building.

Furthermore, the experiments' setup aims to demonstrate the ability of the proposed SLAM in two different loop scenarios. In the first scenario, we ended the loop by revisiting the start place of our scanning (ITC_f1_SmallLoop and ITC_f1_LargeLoop), while in the second scenario, we revisited a place that is not the start or end place of our scanning (FB_f0).

For the experiments, we used min_N_p = 100 for the minimum number of points in a planar feature, max_σ_{p-Pl} = 3 cm for the maximum standard deviation of plane fitting residuals of points and $OBB_{d-threshold}$ = 30 cm for the minimum bounding box's extent. The distance and angle thresholds for the association between the extracted planar segments and the planes in the SLAM map were 10 cm and 3°, respectively. For loop closure detection, these thresholds are set to 3 m and 15°, respectively. The time interval threshold between a pair of planes depends on the loop size. For ITC_f1_SmallLoop dataset, this threshold is set to 25 sec, which covers the interval of 1000 scans, while for the other loops it is set to 125 sec, which cover the interval of 5000 scans.

4.3. Analysis of SLAM performance

As shown in Figures 6 and 7, our SLAM system could cope with both study areas and generate point clouds successfully for all datasets. A 3D animation demonstrating these point clouds is available on video [32]. The final column in Table 1 lists the number of points in each point cloud.

The proposed hypothesis generation of planar structures allows the SLAM to reconstruct arbitrarily oriented planes (slanted). This was evident through the ability of SLAM to map two different staircase environments, namely the central and emergency exit staircases, in the ITC building where several slanted planes are present (Figure 6a&c). Figure 6c clearly shows the slanted planes in the exit staircase. The ability to map stairs and handle the arbitrarily oriented planar structures enables our SLAM to map multi-storey spaces in both the ITC and Fire Brigade buildings (Figure 6a&c and Figure 7b).

The benefits of the proposed hypothesis generation were not limited to handling slanted planes but also extended to reconstructing the rounded walls and circular bar in the Fire Brigade datasets (Figure 7c). Furthermore, the cluttered first floor in the Fire Brigade building was mapped successfully. The most crucial aspect in the testing of the proposed SLAM was to check the robustness in terms of loop closure detection and correction. This was evident through the ability to detect and correct the loop closure in all four different loops (Figure 6a,b&d and Figure 7a). We computed the accumulated

Table 1. General information about captured datasets

Building	Datasets (the terminology used in the paper)	Collection Time (m)	Key characteristics	Number of points
ITC building	ITC_f2f3f2_Loop	5	-Slanted planar structures -Narrow and long corridors -Narrow and closed staircase -Open staircase -Multi-storey space	40×10^6
	ITC_f2f5_StairCase	4	-Slanted planar structures -Narrow and closed staircase -Multi-storey space	32×10^6
	ITC_f1_LargeLoop	7.5	-Narrow and long corridors -Large loop	58×10^6
	ITC_f1_SmallLoop	1	-Small loop	8×10^6
Fire Brigade building	FB_f0	11	-There is a hall that has very large glass walls -Three large fire trucks were parked in the hall. -Small rooms	84×10^6
	FB_f1	6	-Curved walls and bar -Cluttered	48×10^6
	FB_f0f1_Stairs	2	-Straight stairs -Multi-storey space	15×10^6

drift at the end of each loop without any loop closure correction. The results show that our SLAM accumulates a drift of only 0.08% (0.15 m) and $0.006°/m$ (1°) over a 183 m long acquisition trajectory on the ground floor in the Fire Brigade building. The accumulated drift is only high in the areas that have homogeneous, long corridors as is the case for the two large loops in the ITC building. However, the drift does not exceed 0.32% (0.80 m) and $0.013°/m$ (2.6°) over a length trajectory of about 250 m.

The resulting drifts are mainly on the X and Y axes, which represent the moving direction based on the orientation of the scanned area with respect to our model system. The drift in the Z-axis direction is negligible and is smaller than the threshold used for data association, thus the system still sees the previously estimated planes for floor and ceiling in the second visit.

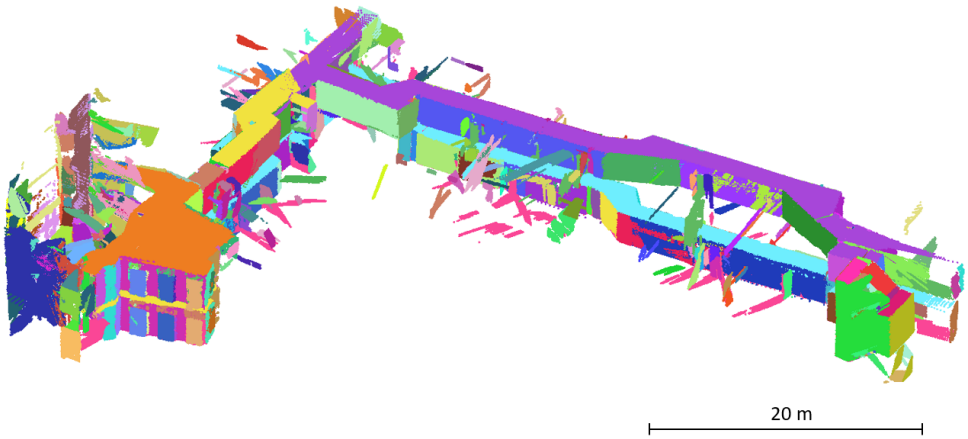
Performing the SLAM algorithm on the first-floor loop (ITC_f1_LargeLoop) and the second-third floor loop (f2f3f2_Loop) in the ITC building was particularly challenging, mainly because of the large loop and the long narrow corridors. This is noticeable as we have the largest misclosure in these two datasets. Misclosures are predominantly caused by positioning errors in the direction of long corridors.

Figure 8 shows the loop closure for the largest loop in our experiments (ITC_f1_LargeLoop) both with and without the proposed loop closure correction. It can be seen that, without loop closure, reobserving places introduces duplicate walls into the map.

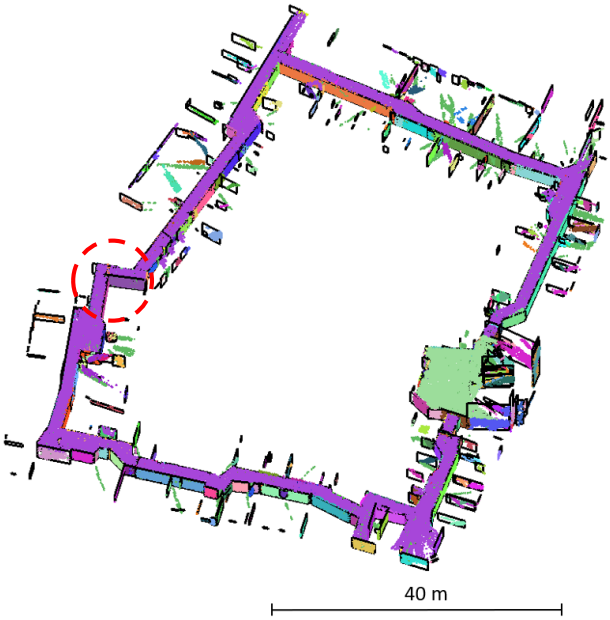
4.4. Cloud to Cloud Comparison

To evaluate the performance of our SLAM system, we compared the generated point clouds to those derived from a commercial MMS (Viametris iMS3D) and a TLS (RIEGL VZ-400).

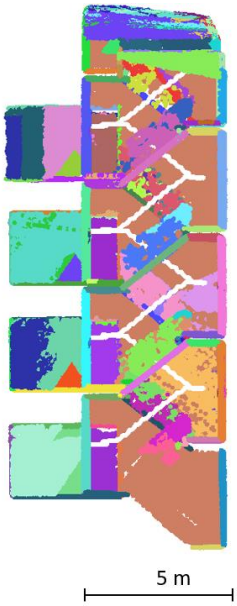
The point clouds have far-away points that were captured in unvisited areas through glass or open doors. As these points will often not be present in both clouds, we have discarded these points from the comparison.



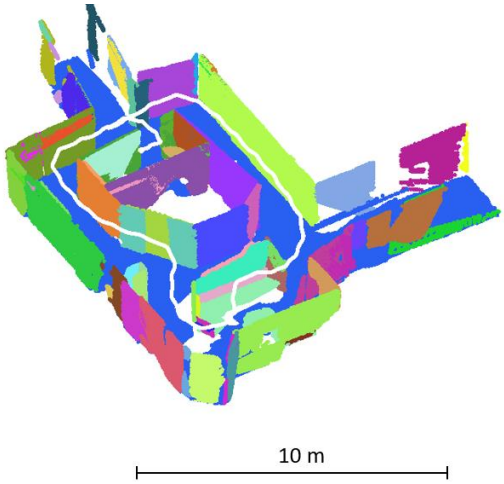
(a) ITC_f2f3f2_Loop



(b) ITC_f1_largeLoop

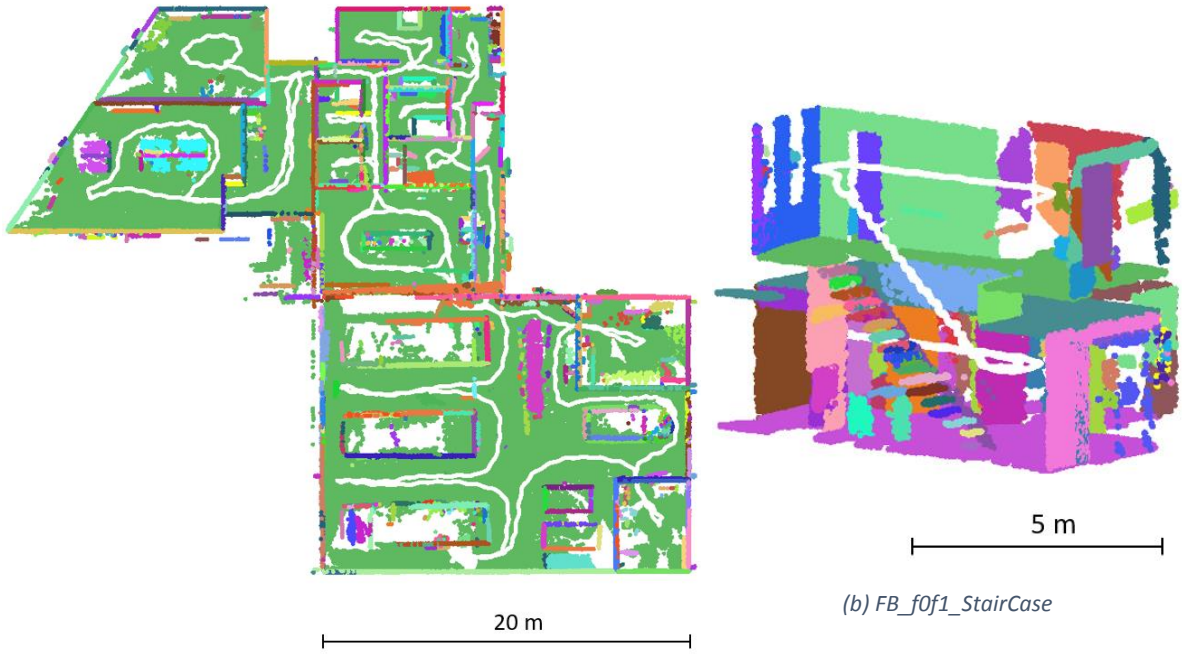


(c) ITC_f2f5f2



(d) ITC_f1_smalLoop

Figure 6. Slanted view of the generated point clouds of the ITC building datasets captured by the ITC-Backpack. Colours show plane association. The black 2D bounding boxes in (b) represent the reconstructed planes in the Global-map and the dashed red circle represents the detected loop closure area after the correction; see Figure 8. Parts of the last two point clouds are not shown in order to show the interior structure with the trajectory (white).

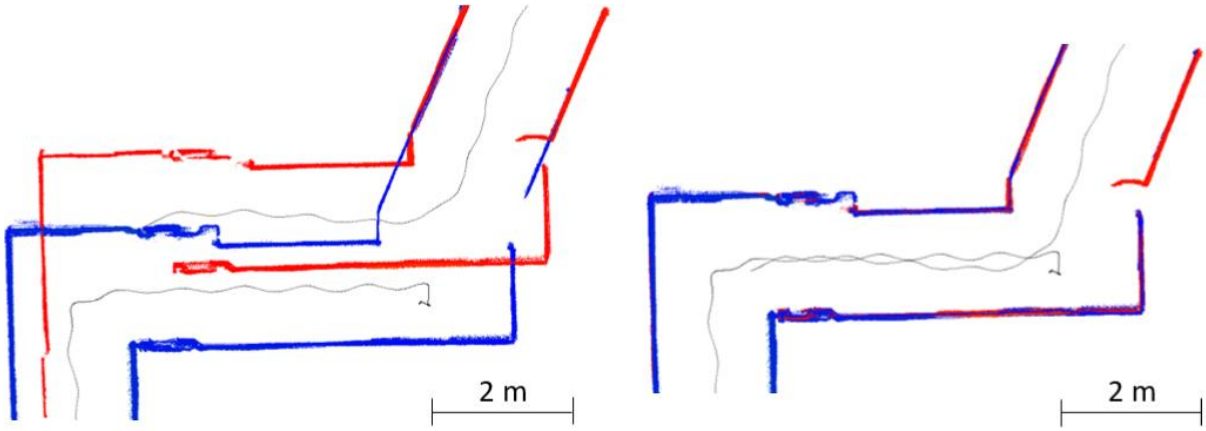


(a) FB_f0

(b) FB_f0f1_StairCase

(c) FB_f1

Figure 7.Top view (a & c) and slanted view (b) of the generated point clouds of the Fire Brigade building datasets captured by the ITC-Backpack. Colours show plane association. The ceiling points in (a) and (b) and parts of the point cloud in (c) are not shown in order to show the interior structure with the trajectory (white). The wavy form of the trajectory follows from a natural walking pace.



(a) Without loop closure *(b) With loop closure*

Figure 8. The first-floor loop in the ITC building (ITC_f1_largeLoop) which is the largest loop in our experiments. (a) Top view of the walls' points with trajectory (black) at the start (blue) and end (red) of the loop without loop closure. (b) The walls' points and the trajectory after loop closure correction.

4.4.1. Comparison against a commercial mobile mapping system

On the same day as our study, the first floor in the Fire Brigade building (FB_f1) was scanned by a commercial mobile mapping system, Viametris iMS3D [33]. For comparison purposes, both our and Viametris's point clouds were registered in the same coordinate system using rigid transformation. The registration process was performed by means of coarse registration and the ICP algorithm included in the open-source free software CloudCompare. The difference between the clouds was undetectable by eye. To quantify the deviation of our point cloud from the Viametris iMS3D cloud, we computed the cloud-to-cloud absolute distances (C2C), also using CloudCompare; see Figure 9. The results show that the majority of the computed distances (about 92%) are less than 3 cm.

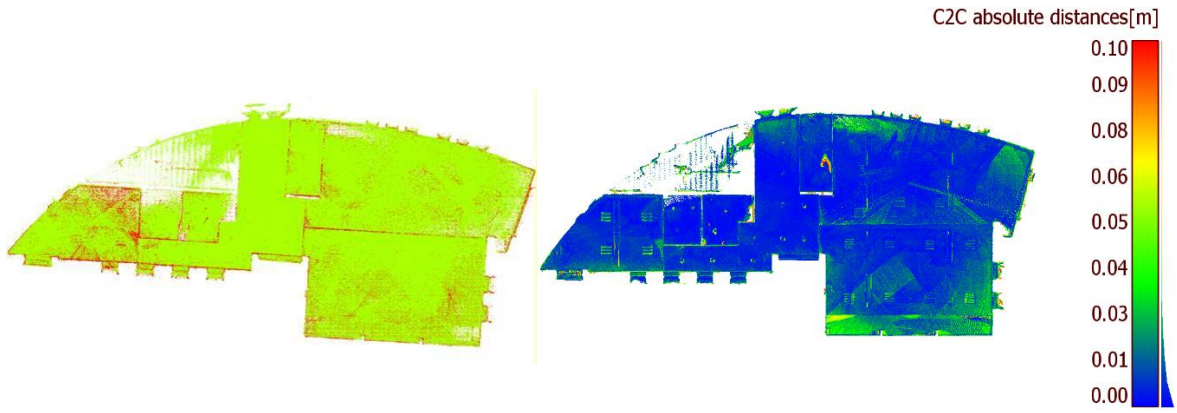


Figure 9. ITC-Backpack and Viametris iMS3D comparison. Left: Our point cloud (red) and iMS3D cloud (green) registered in the same coordinate system. Right: Our point cloud coloured based on the distances to the iMS3D cloud

4.4.2. TLS comparison

Furthermore, we have scanned the second-third floor loop (ITC_f2f3f2_loop), as that is one of the most challenging datasets in the ITC building by RIEGL VZ-400 TLS through a series of individual scans (39 scan positions). The registration of scan positions to generate a single point cloud was performed by means of coarse registration and the ICP algorithm included in the multi-station adjustment plugin within the RiSCAN PRO software accompanying RIEGL TLS. The registration error was less than 1 cm.

As in the previous comparison, the point clouds from our system and TLS scanner were registered in the same coordinate system. Visual inspection shows differences between the clouds. We quantified

the deviation of our point cloud to the TLS cloud used as ground truth, as shown in Figure 10. The histogram of C2C distances shows that 86% of the computed distances are less than 20 *cm*.

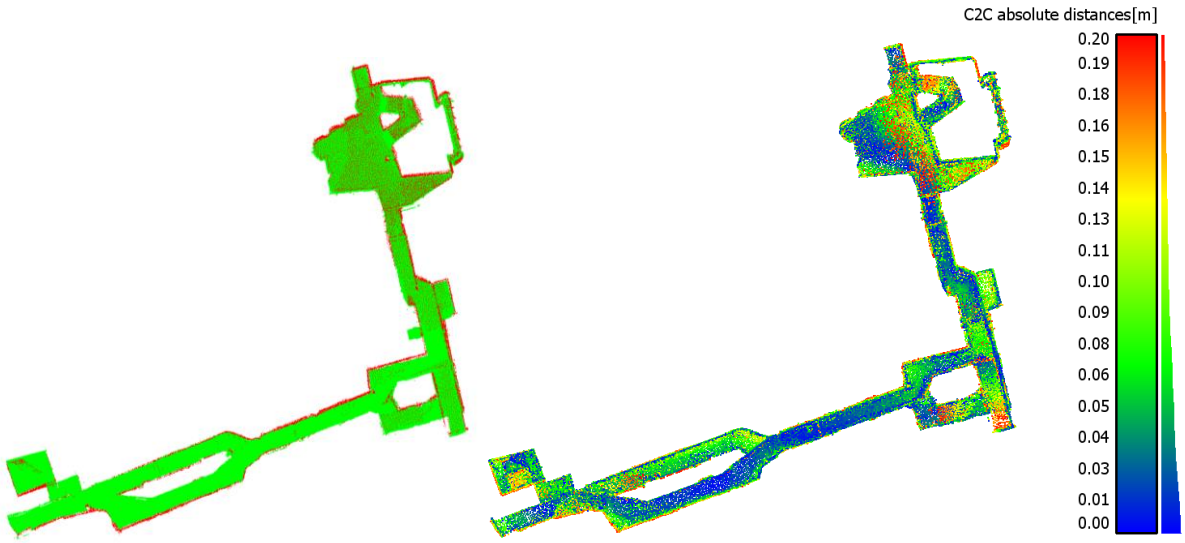


Figure 10. ITC-Backpack and TLS comparison. Left: Our point cloud (red) and TLS cloud (green) registered in the same coordinate system. Right: Our point cloud coloured based on the C2C distances to the TLS cloud

4.5. Discussion and limitations

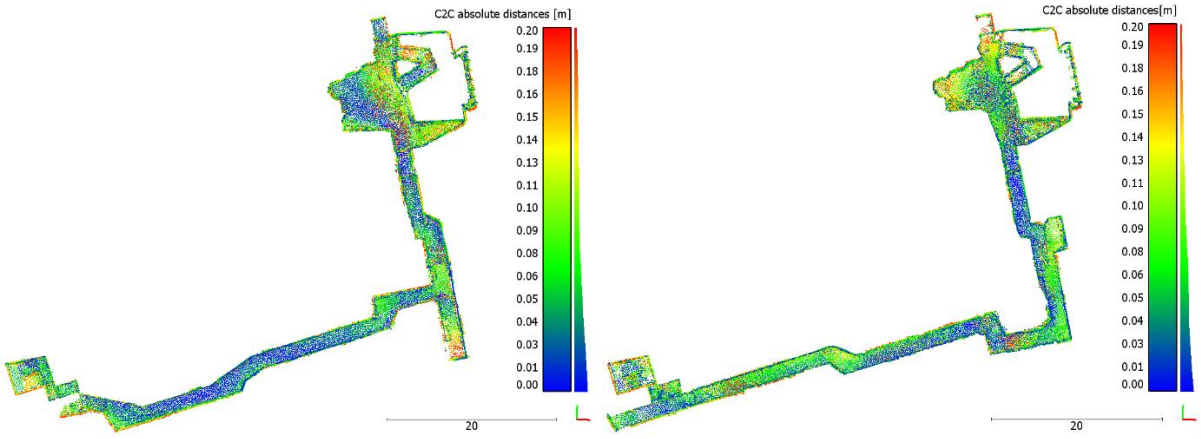
The proposed method has limitations. The detection of a standard-height vertical wall, when observed from far-away e.g. from the other end of a corridor, becomes difficult as it can be observed only within a limited tilt angle i.e. an almost horizontal scan line. In this case, the farther away the observation is made, the more uncertain it is that the observation was made from the wall and not from the floor or the ceiling. Moreover, distinguishing whether a far-away wall is vertical or slanted is also problematic. Furthermore, in some cases, the data on such a wall is thin up to a point that makes the plane extraction impossible. In such a pathological environment, e.g. a long homogeneous corridor, the method will start to drift (as its IMU does) along the corridor, i.e. the direction along which LiDAR observations do not serve to update the pose.

To mitigate this drift and avoid degenerate motions, we slightly relaxed the association distance threshold (by a factor of two) for the linear segments that are extracted from horizontal scan lines and far away from the system in an attempt to assign the segments on the wall at the end of a corridor to the nearest plane that had been previously estimated and most probably represented this wall.

Another limitation is that the state vector formulation of Eq. (18) assumes time-independent autocalibration of sensors. Therefore, we chose to pre-calibrate the IMU and apply fixed biases in pre-processing. The IMU is Xsens MTi-100 and, based on our results, is of high enough quality for this approach. In future work, however, we will investigate whether more map accuracy can be gained if the state vector of Eq. (18) accounts for time-dependent calibration by introducing piece-wise calibration parameters similarly as in [5,12,19].

The proposed loop closure technique corrected the misclosure at the end of all loops in our datasets (Figure 6a,b&d and Figure 7a). As is commonly known, loop closure reduces errors, but also distributes errors over the whole loop to make the data internally consistent. Therefore, we compare against the TLS point cloud, where these errors clearly remain visible (Figure 10). For further investigation of this errors distribution, we separated out the third floor's point cloud from the generated point cloud (ITC_f2f3f2_loop) after loop closure (Figure 6a). Next, we compared the segmented cloud against the corresponding TLS cloud (Figure 11a). The C2C distances between these two clouds demonstrate that about 89% of the computed distances are less than 20 *cm* which is slightly higher than the percentage for the whole loop comparison (Figure 10). Moreover, we ran SLAM separately just on the third floor's data. The C2C distances between the generated point cloud and the corresponding TLS cloud show

that about 84% of computed distances are less than 20 cm which means that the percentage of long distances (> 20 cm) is increased by about 5% if we do not process the whole loop and correct the misclosure. We repeated the test above on the second floor's cloud (Figure 11b) and in contrast to the third floor, the percentage of long distances (> 20 cm) is slightly increased after the loop closure correction. This can be explained as a distribution of the larger errors on the third floor over the entire loop as a result of the loop closure.



(a) *(b)*

Figure 11. Point cloud of part of the third (a) and second (b) floor in the ITC building coloured based on the distances to the corresponding TLS cloud

Introducing the scan trajectory along with the point cloud as supplementary information is advantageous for semantic interpretation [29] and space partitioning of the point cloud [30]. Expanding this supplementary information to cover also the plane features of the SLAM map will likely be also beneficial; and part of our future work. Simultaneously, we will then address a limitation in our current work that the association of leftover points (Section 3.2) may lead to points being incorrectly associated onto an existing plane, before enough points are seen to instantiate a new plane.

5. Conclusions

This article has introduced a novel loop-closing continuous-time LiDAR-IMU Graph-SLAM method to map indoor environments. The SLAM map representation is done by a set of planes. The data association technique assigns detected planar segments as new or onto existing planes, which both can have arbitrary orientations. Regardless of this, a large number of planes still remain either horizontal or vertical in built environment; a fact which we exploit to reduce the number of free parameters in the joint optimization problem, by classifying planes into horizontal, vertical, and slanted classes. Loop closure detection and correction is done by pair-wise planar feature matching, relying on the fact that the planar features are large and hence spatially distinct.

Our results are compared against a commercial mapping system (iMS3D) and the ground truth (RIEGL VZ-400 TLS). Results show that our SLAM system shows comparable performance with the commercial mobile mapping system because 92% of the C2C distances, between the corresponding clouds from both systems, are less than 3 cm. While the deviation is larger to the TLS cloud, 86% of the C2C distances are less than 20 cm, due to the presence of long homogenous corridors in the compared cloud. The proposed method was verified in various scenarios, including scanning multi-storey space, staircases, large and small loops, cluttered areas, and areas surrounded by some rounded walls.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

Acknowledgment

The authors would like to thank the Fire Brigade Haaksbergen for making the building available for data collection. We acknowledge Isabelle Dikhoff and Yahya Ghassoun from the University of Braunschweig for kindly providing the Viametris point cloud.

References

1. Lehtola, V. V.; Kaartinen, H.; Nüchter, A.; Kaijaluoto, R.; Kukko, A.; Litkey, P.; Honkavaara, E.; Rosnell, T.; Vaaja, M.T.; Virtanen, J.P.; et al. Comparison of the selected state-of-the-art 3D indoor scanning and point cloud generation methods. *Remote Sens.* **2017**, *9*, 1–26, doi:10.3390/rs9080796.
2. Maboudi, M.; Bánhidi, D.; Gerke, M. Evaluation of indoor mobile mapping systems. *GFaI Work. 3D North East 2017 (20th Appl. Work. Meas. Model. Process. Anal. 3D-Data)* **2017**, 125–134.
3. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332, doi:10.1109/TRO.2016.2624754.
4. Yu, N.; Zhang, B. An Improved Hector SLAM Algorithm based on Information Fusion for Mobile Robot. Proc. 2018 5th IEEE Int. Conf. Cloud Comput. Intell. Syst. CCIS 2018 2019, 279–284, doi:10.1109/CCIS.2018.8691198.
5. Geneva, P.; Eckenhoff, K.; Yang, Y.; Huang, G. LIPS: LiDAR-Inertial 3D Plane SLAM. *IEEE Int. Conf. Intell. Robot. Syst.* **2018**, 123–130, doi:10.1109/IROS.2018.8594463.
6. Karam, S.; Lehtola, V.; Vosselman, G. Strategies to integrate IMU and LiDAR SLAM for indoor mapping. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2020**, *V-1–2020*, 223–230, doi:10.5194/isprs-annals-V-1-2020-223-2020.
7. Yang, Y.; Geneva, P.; Eckenhoff, K.; Huang, G. Degenerate motion analysis for aided INS with online spatial and temporal sensor calibration. *IEEE Robot. Autom. Lett.* **2019**, *4*, 2070–2077, doi:10.1109/LRA.2019.2893803.
8. Karam, S.; Lehtola, V.; Vosselman, G. Integrating a low-cost MEMS IMU into a laser-based SLAM for indoor mobile mapping. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. - ISPRS Arch.* **2019**, *42*, 149–156, doi:10.5194/isprs-archives-XLII-2-W17-149-2019.
9. Ji, S.; Qin, Z.; Shan, J.; Lu, M. Panoramic SLAM from a multiple fisheye camera rig. *ISPRS J. Photogramm. Remote Sens.* **2020**, *159*, 169–183, doi:10.1016/j.isprsjprs.2019.11.014.
10. Lin, J.; Zheng, C.; Xu, W.; Zhang, F. R2LIVE: A Robust, real-time, LiDAR-Inertial-Visual tightly-coupled state estimator and mapping. **2021**.
11. Grisetti, G.; Kümmerle, R.; Stachniss, C.; W. Burgard. A Tutorial on Graph-Based SLAM. *IEEE Intelligent Transportation Systems Magazine 2.4* **2010**, 31–43.
12. Le Gentil, C.; Vidal-Calleja, T.; Huang, S. IN2LAAMA: Inertial Lidar localization autocalibration and mapping. *IEEE Trans. Robot.* **2020**, *37*, 275–290, doi:10.1109/TRO.2020.3018641.
13. Vosselman, G. Design of an indoor mapping system using three 2D laser scanners and 6 DOF SLAM. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci. 2.3 173.* **2014**.
14. Steder, B.; Grisetti, G.; Burgard, W. Robust place recognition for 3D range data based on point features. *Proc. - IEEE Int. Conf. Robot. Autom.* **2010**, 1400–1405, doi:10.1109/ROBOT.2010.5509401.
15. Bosse, M.; Zlot, R. Place recognition using keypoint voting in large 3D lidar datasets. *Proc. - IEEE Int. Conf. Robot. Autom.* **2013**, 2677–2684, doi:10.1109/ICRA.2013.6630945.
16. He, L.; Wang, X.; Zhang, H. M2dp: A novel 3D point cloud descriptor and its application in loop closure detection. *IEEE Int. Conf. Intell. Robot. Syst.* **2016**, *2016-Novem*, 231–237, doi:10.1109/IROS.2016.7759060.
17. Guo, J.; Borges, P.V.K.; Park, C.; Gawel, A. Local descriptor for robust place recognition using

LiDAR intensity. *arXiv* **2018**, *4*, 1470–1477.
18. Karam, S.; Vosselman, G.; Peter, M.; Hosseinyalamdary, S.; Lehtola, V. Design, calibration, and evaluation of a backpack indoor mobile mapping system. *Remote Sens.* **2019**, *11*, doi:10.3390/rs11080978.
19. Zhou, L.; Koppel, D.; Ju, H.; Steinbruecker, F.; Kaess, M. An Efficient planar bundle adjustment algorithm. *Proc. - 2020 IEEE Int. Symp. Mix. Augment. Reality, ISMAR 2020* **2020**, 136–145, doi:10.1109/ISMAR50242.2020.00035.
20. Gentil, C. Le; Vidal-Calleja, T.; Huang, S. IN2LAMA: INertial lidar localisation and mapping. *Proc. - IEEE Int. Conf. Robot. Autom.* **2019**, *2019-May*, 6388–6394, doi:10.1109/ICRA.2019.8794429.
21. Nikoohemat, S.; Diakité, A.A.; Zlatanova, S.; Vosselman, G. Indoor 3D reconstruction from point clouds for optimal routing in complex buildings to support disaster management. *Autom. Constr.* **2020**, *113*, 103109, doi:10.1016/j.autcon.2020.103109.
22. De La Puente, P.; Rodriguez-Losada, D. Feature based graph SLAM with high level representation using rectangles. *Rob. Auton. Syst.* **2015**, *63*, 80–88, doi:10.1016/j.robot.2014.09.006.
23. Zhang, J.; Singh, S. LOAM : Lidar odometry and mapping in real - time. **2014**, doi:10.15607/RSS.2014.X.007.
24. Cinaz, B.; Kenn, H. Head SLAM - Simultaneous localization and mapping with head-mounted inertial and laser range sensors. *Proc. - Int. Symp. Wearable Comput. ISWC* **2008**, 3–10.
25. Lenac, K.; Kitanov, A.; Cupec, R.; Petrović, I. Fast planar surface 3D SLAM using LIDAR. *Rob. Auton. Syst.* **2017**, *92*, 197–220, doi:10.1016/j.robot.2017.03.013.
26. Zhang, J.; Singh, S. Low-drift and real-time lidar odometry and mapping. *Auton. Robots* **2017**, *41*, 401–416, doi:10.1007/s10514-016-9548-2.
27. Qin, C.; Ye, H.; Pranata, C.E.; Han, J.; Zhang, S.; Liu, M. LINS: A Lidar-Inertial state estimator for robust and efficient navigation. *IEEE Int. Conf. Robot. Autom. (ICRA), pp. 1271-1278* **2020**, 8899–8905.
28. Deschaud, J.E. IMLS-SLAM: scan-to-model matching based on 3D data. *arXiv* **2018**, 2480–2485.
29. Nikoohemat, S.; Peter, M.; Elberink, S.O.; Vosselman, G. Semantic interpretation of mobile laser scanner point clouds in Indoor Scenes using trajectories. *Remote Sens.* **2018**, *10*, doi:10.3390/rs10111754.
30. Elseicy, A.; Nikoohemat, S.; Peter, M.; Elberink, S.O. Space subdivision of indoor mobile laser scanning data based on the scanner trajectory. *Remote Sens.* **2018**, *10*, 1–26, doi:10.3390/rs10111815.
31. Vosselman, G.; Gorte, B.; Sithole, G.; Rabbani, T. Recognising structure in laser scanner point clouds. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2004**, *46*, 33–38.
32. ITC backpack point clouds - YouTube https://www.youtube.com/playlist?list=PLVQwHcKd7n9O_zh_R2NDBh4-P0aVeY8bA.
33. Viametris iMS3D. Available online: http://www.viametris.com/products/ims3d/ (accessed on Nov 20, 2018).