Simple loop closing for continuous 6DOF LIDAR&IMU graph SLAM with planar features for indoor environments

S. Karam*, V. Lehtola, G. Vosselman

Dept. of Earth Observation Science, Faculty ITC, University of Twente, 7514 AE Enschede, the Netherlands

ARTICLE INFO

Keywords:
Graph SLAM
Loop closure
Planar features
Indoor
Point clouds
Mobile mapping
LIDAR
Planar structures hypothesis
IMU
Splines
6DOF

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 (Via-metris 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.

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 (Lehtola et al., 2017; Maboudi et al., 2017). 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 (Cadena et al., 2016) 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 (Yu and Zhang, 2019) 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 (Geneva et al., 2018; Karam et al., 2020), 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 (Yang et al., 2019; Karam et al., 2019), 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. (Ji et al., 2020; Lin et al., 2021; Fossel et al., 2015; Hess et al., 2016). This is in contrast to the offline SLAM that optimizes the trajectory over the entire recorded data such as graph SLAM (Grisetti et al., 2010). 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
advantageous because it makes storage easier and data association performance of our SLAM system by comparing the obtained point construction (Nikoohemat et al., 2020). This kind of representation is also advantageous in real world. Our work is built on (Karam et al., 2020; Karam et al., 2019; He et al., 2016; Guo et al., 2018) 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. These can be deep learned, e.g. (Chen et al., 2020). 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 (Karam et al., 2020; Karam et al., 2019; Vosselman, 2014; Karam et al., 2019). 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 parameterization 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 (Vosselman, 2014) without a need to do IMU pre-integration (Geneva et al., 2018; Zhou et al., 2020) or upsampling to approximate continuous time (Le Gentil et al., 2021; Gentil et al., 2019).

4) Our SLAM system exploits the IMU to predict the pose of a few successive scans (Section 3.1), similar to e.g. (Le Gentil et al., 2021; Gentil et al., 2019). 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 (Karam et al., 2020), briefly introduced in Sections 3.1 and 3.6.

6) Auto-calibration of LIDARs is done exploiting the planar feature geometry as in our previous work (Karam et al., 2019), lately adopted also in (Le Gentil et al., 2021).

The chosen map representation (Vosselman, 2014; Karam et al., 2019) allows for outputting the scanned environments in planar shapes, which is a popular format for the state-of-the-art in indoor 3D reconstruction (Nikoohemat et al., 2020). 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. (Cedena et al., 2016; De La Puente and Rodriguez-Losada, 2015). 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 with Zebedee (Bosse and Zlot, 2009; Bosse et al., 2012) and LOAM (Zhang and Singh, 2014). The scanners are mounted on a wearable platform (backpack) where one scanner is mounted horizontally and the other two scanners have a tilted orientation.

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 (Le Gentil et al., 2021), LIPS (Geneva et al., 2018) and planar bundle adjustment (PBA) (Zhou et al., 2020). 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 (Zhou et al., 2020) 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 (Le Gentil et al., 2021) 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. (De La Puente and Rodriguez-Losada, 2015), methods assuming vertical walls, e.g. (Cinaz and Kenn, 2008) and a planar feature-based SLAM system experimenting with dividing and projecting the 3D LIDAR data onto three image planes before applying planar segmentation (Lenac et al., 2017).

Apart from graph slam methods, local planarity of points is exploited also in real-time SLAM. The formulation in (Zhang and Singh, 2017) to describe point planarity is sufficiently fast to be used with Kalman filter-based estimators and has proven to be successful in (Lin et al., 2021; Zhang and Singh, 2014; Qin et al., 2020). 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 (Deschaud, 2018).

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 Fig. 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 (Fig. 5). The generated point cloud, together with trajectory information, can be used for the semantic interpretation (Nikoohemat et al., 2018) or space partitioning (Elseicy et al., 2018). 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

We adopt the coordinate system of the horizontal scanner as the frame coordinate system ($f$). This frame system is attached to a moving backpack platform. The model coordinate system ($m$) (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 (Vosselman, 2014). We define the direction of the Z-axis as the normal vector of the horizontal ceiling or floor’s plane, and the direction of the X-axis as the vector product of the Z-axis and the normal vector of a wall plane. Then, the direction of the Y-axis is defined by the vector product of the Z-axis and the X-axis. 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-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 (Le Gentil et al., 2021; Gentil et al., 2019).

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 (Vosselman et al., 2004) and fit a plane to the points of each segment. Fig. 2 shows the extracted planes

![Fig. 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.](image-url)
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 \( \sigma_{\text{plane}} \) higher than \( \text{max}_{N_{p}} \sigma_{\text{plane}} \) 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 \( t_{r} \) (\( r = n + 1, \ldots, N \)) from the scan-combination captured in \( N - n = 10 \) time steps, i.e. 0.25 sec; and (ii) SLAM map planes (plane features) estimated in the previous time steps \( t_{p}, p = 1, 2, \ldots, n \).

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 parameterisation

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

\[
\mathbf{n} p - d = 0
\]

(1)

where \( \mathbf{n} = (n_{x}, n_{y}, n_{z}) \) 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 \( (\theta_{h}, \phi_{h}, d_{h}) \) 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 (Fig. 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 \( (x, y, z, \omega, \phi, \kappa) \) are modelled as continuous functions over time using six cubic splines, one for each degree of freedom, (Karam et al., 2019) which guarantees a level of smoothness for the final trajectory and allows for a continuous time representation for individual measurements. For instance, roll \( \omega \) is formulated as follows:

\[
\omega(t) = \sum_{i} a_{\omega,i} B_{i}(t)
\]

(2)

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

The cubic splines, a knot dividing adjacent intervals is placed once every 5 scan lines (i.e. every 0.125 s). Hence, the distance between the knots at a walking speed of 1 m/s will be 12.5 cm only. The modelling error of the trajectory with such a 4th order spline will almost always be below the instrumental noise level of the range sensor (1-2 cm). At both ends of the trajectory, we have a natural spline boundary condition to avoid artefacts from overfitting.

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 \( X_{\text{total}} \) consists of spline coefficients for each node interval \( i \), where \( i \) indexes the M spline intervals, and parameters of all planes, as follows:

\[
X_{\text{total}} = [\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{iM}, \mathbf{P}_{1}, \mathbf{P}_{2}, \ldots, \mathbf{P}_{H}]
\]

(3)

where \( \alpha_{i} = [\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{iM}] \) are modelled as cubic B-splines on interval \( i \) and \( \mathbf{P}_{h} = (\theta_{h}, \phi_{h}, d_{h}) \) is the parametrization of plane \( h = 1, 2, \ldots, H \).

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 3.4.2) 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 \( \mathbf{P}_{h} = (n_{h}, d_{h}) \) equal zero (Vosselman, 2014; Karam et al., 2019). Due to the sensor noise and estimation errors, this distance is not necessarily zero, but a residual \( r_{p} \).

Consequently, for a point \( p_{m}^{n} = (\mathbf{x}^{m}, \mathbf{y}^{m}, \mathbf{z}^{m}) \) in the model coordinate system \( m \), the equation can be formulated as follows:

\[
r_{p}^{n} = n s p_{f}^{n} - d_{s},
\]

(4)

where \( j \) indexes captured LIDAR points.

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

\[
r_{p}^{m} = \cos\phi_{s} \cos\theta_{s} x^{m} + \cos\phi_{s} \sin\theta_{s} y^{m} + \sin\phi_{s} z^{m} - d_{s}
\]

(5)

Accordingly, the plane \( \mathbf{P}_{h} \) is represented as \( \mathbf{P}_{h} = (\theta_{h}, \phi_{h}, d_{h}, \mathbf{OBB}_{h}) \) where \( \mathbf{OBB}_{h} \) contains information about the 2D bounding box extents. The mapping system records the point \( p_{t} \) at time \( t \) in its platform coordinate system \( f \), which is later transformed to the model system by:

\[
\mathbf{p}_{f}^{m} = R_{f}^{m}(t)\mathbf{p}_{t}^{f} + \mathbf{v}(t)
\]

(6)
where $R^m_f(t_j)$ 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 $p_i$ has its own timestamp $t_i$, our SLAM transforms the point $p_i$ to the model system with rotation $R^m_f(t_j)$ and translation $v(t_j)$ parameters derived from the pose splines at its timestamp $t_j$ as follows:

$$R^m_f(t_j) = R \left( \sum_i a_{ij} B(t_j) \sum_i \Delta \alpha_{ij} B(t_j) \sum_i \Delta \alpha_{ij} B(t_j) \right)$$

$$v(t_j) = \left( \sum_i a_{ij} B(t_j) \sum_i \Delta \alpha_{ij} B(t_j) \sum_i \Delta \alpha_{ij} B(t_j) \right) \left( \right)$$

(7)

(8)

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

$$r^i_f = n_i \left[ R^0_i(t_j)p_i + v_i(t_j) \right] - d_i$$

(9)

After linearizing Eq. (9) with respect to the unknown pose spline coefficients and unknown plane parameters (Vosselman, 2014) using Taylor-series expansion, the residual $r^i_f$ for LiDAR observations is

$$n_i^0 \left[ \left( R^0_i(t_j) p_i' + v_i(t_j) \right) \right] - d_i^0 =$$

$$n_i^0 \left[ \frac{\partial R^0_i(t_j)}{\partial p_i} \sum_i \Delta \alpha_{ij} B(t_j) \right] -$$

$$n_i^0 \left[ \frac{\partial R^0_i(t_j)}{\partial t_j} \sum_i \Delta \alpha_{ij} B(t_j) \right] -$$

$$n_i^0 \left[ \frac{\partial R^0_i(t_j)}{\partial k} \sum_i \Delta \alpha_{ij} B(t_j) \right] -$$

$$n_i^0 \left[ \sum_i \Delta \alpha_{ij} B(t_j) \right] - n_i^0 \left[ \sum_i \Delta \alpha_{ij} B(t_j) \right] - n_i^0 \left[ \sum_i \Delta \alpha_{ij} B(t_j) \right] -$$

$$+ [\sin \theta_i \cos \phi_i \cos \theta_i \cos \phi_i] \left( R^0_i(t_j)p_i' + v_i(t_j) \right) \Delta \theta_i -$$

$$+ [\cos \theta_i \sin \phi_i \sin \theta_i \sin \phi_i] \left( R^0_i(t_j)p_i' + v_i(t_j) \right) \Delta \phi_i + \Delta d_i$$

(10)

where the upper index 0 denotes the approximate value, $\Delta \alpha_{ij}$, $\Delta \alpha_{ij}$, $\Delta \alpha_{ij}$, $\Delta \alpha_{ij}$, $\Delta \alpha_{ij}$, $\Delta \alpha_{ij}$, $\Delta \alpha_{ij}$, $\Delta \alpha_{ij}$, $\Delta \alpha_{ij}$ are the unknown increments of the pose splines coefficients on interval $i$ and $\Delta \theta_i$, $\Delta \phi_i$, $\Delta d_i$ are the unknown increments of the parameters of plane $p_i$.

Eq. (10) is written in general form, i.e. for arbitrarily oriented planes for which all three increments $\Delta \theta$, $\Delta \phi$, $\Delta d$ 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. (2020) 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 $\tilde{V}^m_{imu} = (\dot{\omega}_{imu}, \ddot{\omega}_{imu}, \dot{\omega}_{imu})$ and acceleration $\tilde{T}^m_{imu} = \left( \ddot{X}_{imu}, \ddot{Y}_{imu}, \ddot{Z}_{imu} \right)$ along all three axes over time, we formulate six IMU observation equations at each IMU timestamp $t_i$, Eq. (14) and Eq. (15). The IMU observation equations have been described in (Karam et al., 2020), 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 $\ddot{T}^m_f = \left( \ddot{X}^m_f, \ddot{Y}^m_f, \ddot{Z}^m_f \right)$.

$$\ddot{T}^m_f(t_i) = R^m_f(t_i)\ddot{T}_{imu}(t_i) - \left( 0 \right) = r_f$$

(11)

where $R^m_f = R^m_f(\dot{\theta}_f, \phi_f)$ 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 $r_f$ in this paper.

$$\ddot{T}^m_f(t_i) - R^m_f(t_i) R^s_{imu}(t_i) + \left( 0 \right) = r_f$$

(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 $\dot{V}^m_f = (\dot{\phi}^0_f, \dot{\theta}^0_f, \dot{\phi}^0_f)$ after the rotation to the IMU sensor system.

$$V^s_{imu}(t_i) - R^s_f(t_i) S^s_{imu}(t_i) V^m_{imu}(t_i) = r_v$$

(13)

where $r_v$ is the angular velocity residual at $t_i$, $R^s_f = (R^s_f)^T$ and $S^s_m = \left( \begin{array}{ccc} \cos \theta_s \cos \phi_s \sin \phi_s & 0 & f \\ -\cos \theta_s \sin \phi_s \sin \phi_s & \cos \phi_s & 0 \\ \sin \theta_s & 0 & 1 \end{array} \right)$ is the transformation matrix from the model system to the frame system; see (Karam et al., 2020) 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 $X(t) = \sum_i \Delta \alpha_{ij} B_i(t)$, the acceleration X spline becomes $X(t) = \sum_i \Delta \alpha_{ij} B_i(t)$, where $\Delta \alpha_{ij}$ is the X spline coefficient to be estimated on interval $i$. For the rotation angle $\omega$ spline $\omega(t) = \sum_i \Delta \alpha_{ij} B_i(t)$, the angular velocity $\dot{\omega}$ spline becomes $\dot{\omega}(t) = \sum_i \Delta \alpha_{ij} B_i(t)$, where $\Delta \alpha_{ij}$ is the X 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.
\[
\begin{align*}
\bar{T}_p^n(t_u) - R_p^n(t_u)R_f^s\bar{T}_{imu}^s(t_u) + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} &= - \begin{bmatrix} \sum_i \Delta\alpha_{i,1} \hat{B}_i(t_u) \\ \sum_i \Delta\alpha_{i,2} \hat{B}_i(t_u) \\ \sum_i \Delta\alpha_{i,3} \hat{B}_i(t_u) \end{bmatrix} \\
&+ \frac{\partial R_p^n(t_u)}{\partial p} R_f^s \bar{T}_imu^s(t_u) \sum_i \Delta\alpha_{i,0} \hat{B}_i(t_u)
\end{align*}
\] (14)

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 (Karam et al., 2019). Consequently, the total state to be estimated \(\mathbf{x}_{total}\) in Global_SLAM becomes

\[
\mathbf{x}_{total} = [α_1, α_2, ⋯, α_M, P_1, P_2, ⋯, P_M, μ, \Psi]
\] (18)

where \(μ\) contains the intrinsic sensor calibration parameters. In principle we want to estimate the scale factor for the range, scale factor for the scanning direction and a range offset for all three scanners. But we cannot estimate scale factors for all three scanners at the same time because then we cannot determine the scale of the resulting point cloud. Therefore, we fix the range scale factor of one scanner and \(μ\) becomes 8 \(\times\) 1 vector in the total state. \(Ψ\) contains the parameters describing the relative poses, 3 rotations and 3 translations, of each titled scanner to the frame system (the sensor system of the horizontal scanner). In normal walking mode of the operator, the horizontal scanner scans in an approximately horizontal plane. In the case of capturing only vertical walls, the vertical offset parameter between this scanner and the other two cannot be estimated. Therefore, we need to fix this parameter and \(Ψ\) becomes 11 \(\times\) 1 vector.

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 \(\mathbf{x}_{total}\) 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 Fig. 3) and the second group has been reconstructed during the second visit (coloured blue in Fig. 3).

Therefore, in the Global-SLAM phase (see Fig. 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 Fig. 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.

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

Fig. 4 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) (Karam et al., 2019). 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
Fig. 5. 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 Fig. 7. Parts of the last two point clouds are not shown in order to show the interior structure with the trajectory (white). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)
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).

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 trajectory on the ground floor in the Fire Brigade building. 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 (Fig. 5a,b and Fig. 6a). We computed the accumulated drift at the end of each loop without any loop closure correction. The results show that our SLAM accumulates a drift of only 0.06% (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 0.3% over a 25 sec period, 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 Figs. 5 and 6, 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 (ITC backpack point clouds). Our data is openly available in the robot operating system (ROS) format.

The benefits of the proposed hypothesis generation were not limited to handling slanted planes but also extended to reconstructing the rounded walls and the top floor in the Fire Brigade datasets (Fig. 6c). The cluttered first floor in the Fire Brigade building was mapped successfully.

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 (Fig. 5a,b and Fig. 6a). We computed the accumulated drift at the end of each loop without any loop closure correction. The results show that our SLAM accumulates a drift of only 0.06% (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 0.3% over a 25 sec period, 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.

### 4.4. General information about captured datasets

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

Fig. 7 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.

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 (Viametris iMS3D). 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 (CloudCompare, 2021). 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 (CloudCompare, 2021); see Fig. 8. The results show that the majority of the computed distances (about 92%) are less than 3 cm.

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

Fig. 7. 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. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fig. 8. 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. Note that the Viametris point cloud has an average surface density 280 points/m², while our cloud has an average surface density 145 points/m². (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)
shown in Fig. 9. The histogram of C2C distances shows that 86% of the computed distances are less than 20 cm.

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 (Geneva et al., 2018; Le Gentil et al., 2021; Zhou et al., 2020).

The proposed loop closure technique corrected the misclosure at the end of all loops in our datasets (Fig. 5a, b&d and Fig. 6a). 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 (Fig. 9). 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 (Fig. 5a). Next, we compared the segmented cloud against the corresponding TLS cloud (Fig. 10a). 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 (Fig. 9). 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

Fig. 9. 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. Note that the TLS point cloud has an average surface density 400 points/m², while our cloud has an average surface density 120 points/m². (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fig. 10. 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.
repeated the test above on the second floor’s cloud (Fig. 10b) 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.

Registered 3D point clouds of piecewise planar indoor environments are beneficial for automatic 3D reconstruction (Xiao and Furukawa, 2014; Pintore et al., 2020). In our methodology, introducing the scan trajectory along with the 3D point cloud as supplementary information is advantageous for semantic interpretation (Nikoohemat et al., 2018) and space partitioning of the point cloud (Elseicy et al., 2018). 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.

For direct method comparison purposes, we observe that none of the closest works to ours offers open-source codes, i.e. planar SLAMs (Geneva et al., 2018; Le Gentil et al., 2021; Zhou et al., 2020); so we analyse the capabilities of a not-so-close method, Cartographer\(^\text{2}\), to process the data of our backpack. From the horizontal scanner data, the 2D Cartographer (Hess et al., 2016) can produce the floor map (see Fig. 11), but as it does not estimate 6DoF sensor poses, it cannot be used to generate a high quality 3D point cloud. The 3D Cartographer is not robust enough to run with the data from the three scanners, as it needs overlaps between scans in all dimensions to match scans. Intuitively, our proposed method using planar features is more robust to scarcer input data than the method that relies on point-to-point overlap matching, because our method connects the consecutive scan lines on e.g. a wall to one another without the need for overlap.

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.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

We thank Mr. Holger Rapp from the Google cartographer team for the exchange of correspondence. The authors would like to thank also 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


\(\text{2} \) https://google-cartographer-ros.readthedocs.io

---

Fig. 11. The generated 2D maps by Google Cartographer of FB_f1 (a) and ITC_f1_largeLoop (b) datasets (see Fig. 6c and 5b).