Received: 2 March 2020 | Revised: 25 October 2021 | Accepted: 15 February 2022

DOI: 10.33012/navi.527

ORIGINAL ARTICLE

AGPC-SLAM: Absolute Ground Plane Constrained 3D Lidar SLAM

Weisong Wen | **Li-Ta Hsu**

Hong Kong Polytechnic University

Correspondence

Li-Ta Hsu

Email: lt.hsu@polyu.edu.hk

Abstract

3D lidar-based simultaneous localization and mapping (SLAM) is a well-recognized solution for mapping and localization applications. However, the typical 3D lidar sensor (e.g., Velodyne HDL-32E) only provides a very limited field of view vertically. As a result, the vertical accuracy of pose estimation suffers. This paper aims to alleviate this problem by detecting the absolute ground plane to constrain vertical pose estimation. Different from the conventional relative plane constraint, this paper employs the absolute plane distance to refine the position in the z-axis and the norm vector of the ground plane to constrain the attitude drift. Finally, relative positioning from lidar odometry, constraint from ground plane detection, and loop closure are integrated under a proposed factor graph-based 3D lidar SLAM framework (AGPC-SLAM). The effectiveness is verified using several data sets collected in Hong Kong.

Keywords

dynamic object, ground plane constraint, lidar SLAM, urban canyons

1 | INTRODUCTION

Accurate mapping and localization (Dill & Uijt de Haag, 2016) are significant for autonomous systems such as autonomous driving vehicles (ADVs; Huang et al., 2019) and indoor mobile robotics (Hess et al., 2016). Great efforts have been devoted to achieving accurate simultaneous localization and mapping (SLAM) using 3D light detection and ranging (lidar; Hess et al., 2016) sensors due to their robustness compared with the vision-based SLAM methods (Qin et al., 2018). Vision-based SLAM based on a passive sensor like a camera can be sensitive to illumination and viewpoint changes. Conversely, an active sensor like 3D lidar can provide distance measurements for surrounding environments which are invariant to illumination. The outstanding robustness and precision make 3D lidar an indispensable sensor for large-scale mapping and localization.

3D lidar-based SLAM (Koide et al., 2019; Lin & Zhang, 2020; Shan & Englot, 2018; Wen et al., 2020; Zhang & Singh, 2014; Zhao et al., 2019) has been extensively studied over past decades. In general, the 3D lidar SLAM algorithm can be gracefully divided into two parts, the front-end (Grisetti et al., 2010) and the back-end (Grisetti et al., 2010). The front-end focuses on point cloud registration (Pang et al., 2019; Saarinen et al., 2013). The back-end integrates multiple constraints, such as

positioning from the front-end, positioning from the global navigation satellite system (GNSS; Dow et al., 2009; Shetty & Gao, 2019), and loop closure (Magnusson et al., 2009).

The accuracy of point cloud registration, which is the major part of the front-end, dominates the performance of 3D lidar SLAM. Therefore, numerous works have been studied to improve the point cloud registration process, such as the iterative closest point (ICP; Kuramachi et al., 2015), the normal distribution transform (NDT; Magnusson et al., 2007), and lidar odometry and mapping (LOAM; Zhang & Singh, 2017). The LOAM algorithm obtained top accuracy in the KITTI data set odometry benchmark (Geiger et al., 2012) until December 2020 due to its feature extraction and impressive data association structure. Unlike the point-wise registration methods (e.g., ICP and NDT), the LOAM algorithm extracts meaningful edge and planar features from raw 3D point clouds, leading to lower computational load and decreased sensitivity to the local minimums. Instead of simply relying on finding the transformation between consecutive frames of point clouds (Low, 2004; Pang et al., 2019; Saarinen et al., 2013) via scan-to-scan manner, the LOAM algorithm decouples the registration problem into two parts, coarse odometry and fine mapping.

First, the coarse odometry conducts scan-to-scan matching of the edge and planar features, respectively, to estimate a coarse relative transformation. Second, the innovative fine-mapping process is conducted to map the current frame of point clouds to the continuously generated global map (scan-to-map) based on an initial guess derived by the coarse odometry. The mapping process helps to mitigate the accumulated relative drift from lidar odometry in the first step. In short, the LOAM algorithm obtains better performance compared with the listed ICP and NDT. However, due to the limited field of view (FOV, typically $+10° \sim -30°$ vertically, 0~360° horizontally) for typical 3D lidar, the available features in the vertical direction are significantly fewer than those in the horizontal direction. As a result, inevitably, the 3D lidar SLAM can drift in the vertical direction. To find out the major challenge of 3D lidar SLAM in urban canyons, we extensively evaluated the performance of 3D lidar SLAM in diverse urban canyons (Wen et al., 2018). The results showed that the vertical drift due to a limited FOV of 3D lidar was one of the major problems to be solved for the application of 3D lidar SLAM in urban canyons.

According to our evaluations (Wen et al., 2018), in numerous urban scenarios, it was shown that the ground surface is usually available for ground vehicles installed with 3D lidar. Furthermore, the 3D lidar scanning could provide sufficient ground points even in dense traffic scenes that make ground surface detection possible. Inspired by this remarkable feature, this paper proposes to detect and apply the *Absolute Ground Plane Constraint* (AGPC) to constrain the state of the vehicle to further mitigate the vertical drift. In other words, the AGPC is employed to improve the geometry of the constraint. To mitigate the overall drift over time, the loop closure detection method is applied to exploit the loop closure constraint. Finally, a factor graph-based 3D lidar SLAM framework (AGPC-SLAM) is proposed to fuse relative positioning from LiDAR odometry, constraints from AGPC, and loop closure.

The major contributions of this paper are listed as follows:

- We propose to exploit the AGPC to mitigate the vertical drift of 3D LiDAR SLAM.
- We propose a general 3D SLAM framework (AGPC-SLAM) to integrate the constraint of relative positioning from lidar odometry, the AGPC, and the

constraint from the loop closure. The proposed AGPC-SLAM is a general framework that can easily fuse additional information from sensors such as the GNSS receiver and magnetometers.
- The proposed method is validated using two challenging large-scale data sets collected in Hong Kong. We believe that the proposed method can have a positive impact on both the academic and industrial fields.

The remainder of the paper is structured as follows. The related work is reviewed in Section 2. Then, the overview of the proposed method is introduced in Section 3. The detail of the methodology is introduced in Section 4 before the experimental evaluation presented in Section 5. The conclusion and future work are given in Section 6.

2 | RELATED WORK

To handle this problem, the LeGO-LOAM (Shan & Englot, 2018) method proposes optimizing z-axis related states based on planar features. The drift in the z-axis and pitch angle is slightly mitigated compared with the LOAM algorithm (Zhang & Singh, 2017). However, as the vertical states are estimated relatively concerning the planar points, the drifted error can still accumulate over time with the vertical positioning error reaching more than 10 meters (Shan & Englot, 2018) in the evaluated data set.

Recently, the authors extended their LeGO-LOAM to lidar/inertial (Shan et al., 2020) integration to mitigate the overall drift where the high-frequency pose estimation from inertial measurement unit (IMU) pre-integration was used as the initial guess of the mapping process. Meanwhile, the motion distortion was handled with the help of the IMU pre-integration. However, improvement relied on the cost of the IMU. In other words, the additional IMU could not essentially solve the problem of weak constraint in the vertical direction.

The recent work of the team from the Hong Kong University of Science and Technology (HKUST; Ye et al., 2019) proposed to tightly integrate lidar and IMUs to mitigate the overall drift. The tight integration scheme could effectively improve the geometry of the constraints arising from the raw point clouds. With the help of inertial measurements, the overall accuracy was improved compared to standalone lidar SLAM. According to their evaluation, the vertical drift was improved compared with the LOAM. However, the performance relied on the quality of the applied IMU, and the tightly coupled integration process introduced a significantly higher computational load. Meanwhile, the problem of vertical drift was still unsolved. Instead of simply basing their work on relative positioning, the work (Zheng et al., 2019) utilized the Global Positioning System (GPS) to mitigate the drift of lidar SLAM. Nevertheless, the performance of GPS solutions relies heavily on environmental conditions, and the high-rise buildings in urban canyons significantly degrade their performance leading to large positioning errors (Wen et al., 2019a, 2019b). Similar works were also done in Chang et al. (2020) and He et al. (2020).

The work by Lin and Zhang (2020) included loop closure into LOAM to further reduce drift. Unfortunately, the loop closure was difficult to detect due to the distinct vertical drift. Interestingly, the work of Zuo et al. (2019) made use of both camera and lidar to derive improved odometry accuracy. The camera could help the standalone lidar odometry to survive in a sparse area because the camera could capture texture information. Despite this, our previous work (Bai et al., 2020)

showed that the visual-based positioning shared a similar drawback of being sensitive to dynamic objects in urban canyons. In short, the existing work tends to employ additional sensors to improve the overall accuracy of 3D lidar SLAM, which can only partially reduce the speed of the vertical drift (Shan et al., 2020; Ye et al., 2019) or rely on the GNSS positioning accuracy (Chang et al., 2020; He et al., 2020; Zheng et al., 2019).

3 | OVERVIEW OF THE PROPOSED METHOD

3.1 | Notations and Coordinates

Matrices are denoted as uppercase bold letters. Vectors are denoted as lowercase bold letters. Variable scalars are denoted as lowercase italic letters. Constant scalars are denoted as lowercase letters. To make the proposed framework clearer, the following notations are defined and followed by the rest of this paper.

- The local world frame $(\mathrm{W}\{X^W, Y^W, Z^W\})$ is fixed to the world at the start point of the vehicle, which can also be fixed to the GNSS absolute frame when GNSS is available.
- The local base frame $(\mathrm{L}\{X^L, Y^L, Z^L\})$ originates at the starting point of the local lidar odometry.
- The vehicle body frame $(\mathrm{B}\{X^B, Y^B, Z^B\})$ is fixed at the center of the 3D lidar sensor.

The considered coordinate system is shown in Figure 1. $\mathbf{T}_B^L$ denotes the transformation from the body frame to the local base frame, encoding the information of the lidar odometry during the experiment. The transformation $\mathbf{T}_L^W$ denotes the transformation between the local base frame and the local world frame, which is the drift compensation estimated by the additional constraints (e.g., the loop closure constraints and the AGPC constraint). In other words, the local-based

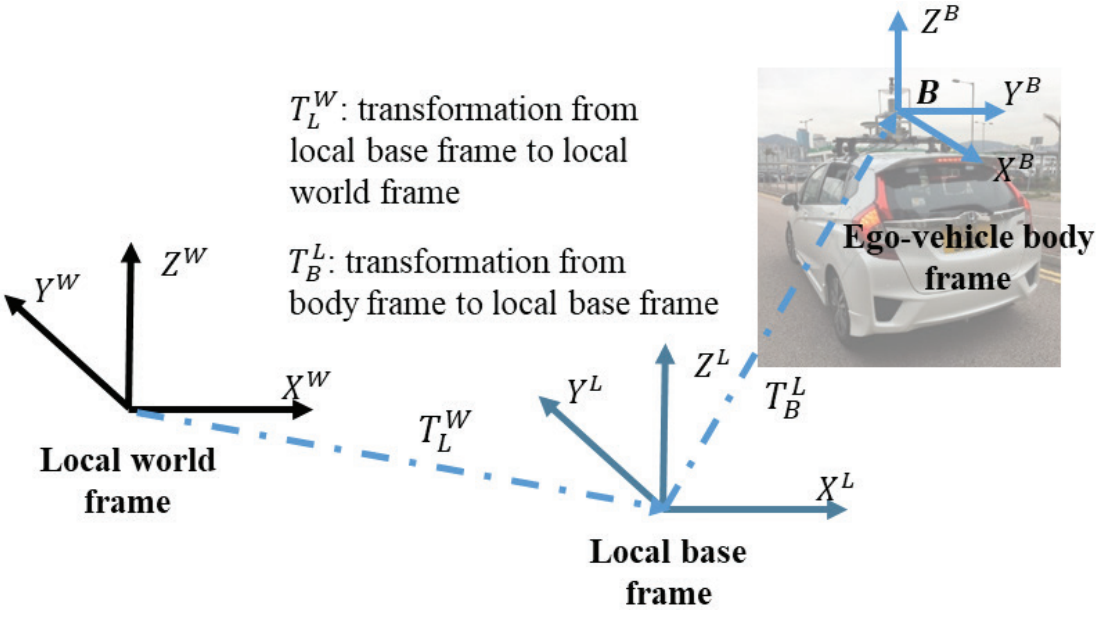

FIGURE 1 Overview of the applied coordinate systems

frame is fixed on the local world frame if there are no ground and loop closure constraints. Note that the definition of the coordinates is referred to in the work of Mascaro et al. (2018) which is commonly used in the SLAM field with loop closure constraint.

Therefore, the state of the ego-vehicle at epoch k is encoded by the transformation from the body frame to the local world frame, $\mathbf{T}_{B,k}^{W}$, as follows:

$$\begin{aligned} \mathbf{T}_{B,k}^{W} &= (\mathbf{p}_{k}^{W}, \mathbf{q}_{k}^{W})^{T} \\ \text{With } \mathbf{T}_{B,k}^{W} &= \boldsymbol{T}_{L,k}^{W} \boldsymbol{T}_{B,k}^{L} \end{aligned} \tag{1}$$

where the $\mathbf{p}_{k}^{W}$ represents the position and the $\mathbf{q}_{k}^{W}$ represents the orientation in the local world frame with quaternion form.

3.2 | Overview of the Proposed AGPC-SLAM

The overview of the proposed AGPC-SLAM framework is shown in Figure 2. The input of the AGPC-SLAM is the 3D lidar point cloud ($\mathbf{S}_k$, the raw point cloud at epoch k). The outputs of the proposed AGPC-SLAM are the vehicular trajectory and the consistent point cloud map.

The AGPC-SLAM framework can be divided into two major parts, the front-end (the light green shaded boxes in Figure 2), and the back-end (the light blue shaded boxes in Figure 2). The light red shaded box denotes the AGPC proposed in this paper. Finally, the constraints from the front-end and the AGPC are integrated using factor graph optimization in the back-end. The details of the major parts are presented in the following sections.

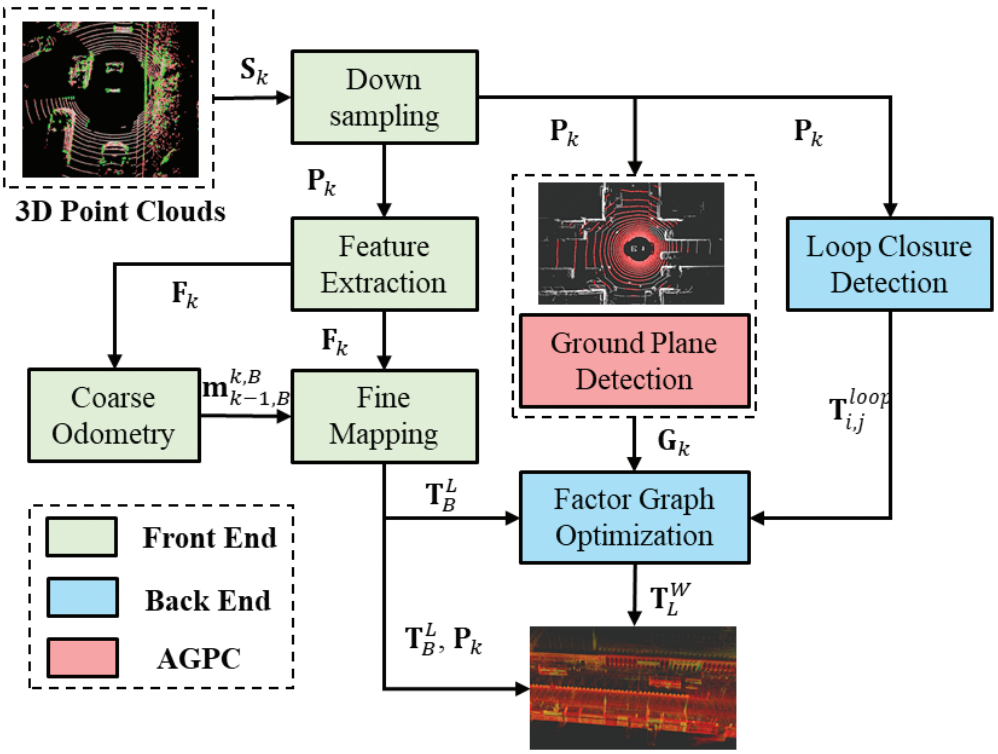

FIGURE 2 Overview of the proposed AGPC-SLAM; the input is the 3D point cloud from 3D lidar. The outputs include the points map and pose estimation of vehicular trajectory.

4 | METHODOLOGY

4.1 | Front-End

The performance of the front-end is dominated by lidar odometry whose objective is to make the best use of the consecutive point clouds to estimate the relative motion. Lidar odometry is implemented based on the LOAM algorithm proposed in Zhang and Singh (2017) in which three steps are included for lidar odometry: feature extraction, coarse odometry, and fine mapping. Although lidar odometry, itself, is not a contribution of this paper, we still briefly present the formulation for the sake of completeness.

4.1.1 | *Feature Extraction*

Different from the point-wise registration method (e.g., NDT and ICP), the feature definition and extraction proposed in Zhang and Singh (2014) explores the representative primitives before the data association process. The input of the feature extraction is the $\mathbf{P}_k\{\mathbf{P}_{k,1}, \mathbf{P}_{k,2}, \ldots, \mathbf{P}_{k,i}, \mathbf{P}_{k,N}\}$. The variable N denotes the number of points inside a frame of point cloud. A point is classified as a planar or an edge point depending on the roughness of its local region. The roughness of the local region is determined as follows (Zhang & Singh, 2017):

$$c_{k,i} = \frac{1}{|\mathbf{S}|\cdot||\mathbf{P}_{k,i}||}\left|\left|\sum_{j\in\mathbf{S}, j\neq i}(\mathbf{P}_{k,j} - \mathbf{P}_{k,i})\right|\right| \tag{2}$$

where $c_{k,i}$ represents the roughness of the point. The variable $\mathbf{S}$ denotes the small local region near the given point $\mathbf{P}_{k,i}$ and, usually, five points are involved in the local region. $\mathbf{P}_{k,j}$ indicates the point belonging to the local region $\mathbf{S}$. If the calculated roughness is larger than a pre-determined threshold t_c, the point is classified as an edge point. The point with roughness that is smaller than the threshold is classified as a planar point. The output of the feature extraction process is the feature set $\mathbf{F}_k\{\mathbf{F}_k^p, \mathbf{F}_k^E\}$, where the $\mathbf{F}_k^p$ and $\mathbf{F}_k^E$ represent the feature set containing all the planar and edge points, respectively. Meanwhile, the authors (Zhang & Singh, 2017) also proposed careful feature point selection strategies to avoid unreliable points. For example, points that are on the boundary of the occluded regions should not be selected as those points can be unobservable in the coming epochs. Meanwhile, the points that lie on local planar surfaces that are roughly parallel to the lidar beams should not be selected as well. We strictly follow these strategies to select feature points from the raw 3D point clouds.

4.1.2 | *Coarse Odometry*

Based on the features extracted in the last section, coarse odometry is performed to efficiently estimate the relative motion between consecutive frames of point clouds. The relative motion is calculated by conducting point-to-edge and point-to-plane scan-matching. In short, the objective is to find the corresponding features for points in $\mathbf{F}_k\{\mathbf{F}_k^p, \mathbf{F}_k^E\}$ from the feature points set $\mathbf{F}_{k-1}\{\mathbf{F}_{k-1}^p, \mathbf{F}_{k-1}^E\}$. Detailed steps can be found in Zhang and Singh (2017). We formulated the point cloud registration process as follows using:

$$\mathbf{m}_{k-1,B}^{k,B} = PCR(\mathbf{F}_k\{\mathbf{F}_k^p, \mathbf{F}_k^E\}, \mathbf{F}_{k-1}\{\mathbf{F}_{k-1}^p, \mathbf{F}_{k-1}^E\}) \tag{3}$$

where *PCR* denotes the point cloud registration function. The output of the point cloud registration process is the coarse relative motion, denoted by $\mathbf{m}_{k-1,B}^{k,B}$. Note that $\mathbf{m}_{k-1,B}^{k,B}$ is the motion between the frame $k-1$ and k. Since the $\mathbf{m}_{k-1,B}^{k,B}$ is estimated based on scan-to-scan matching, the efficiency is guaranteed. Meanwhile, the coarse motion estimation is applied as the initial guess for the fine mapping in the next section.

4.1.3 | *Fine Mapping*

To refine the relative motion estimation, the fine-mapping process is applied to refine the $\mathbf{m}_{k-1,B}^{k,B}$. The principle of the mapping process is that the extracted $\mathbf{F}_k\{\mathbf{F}_k^p, \mathbf{F}_k^E\}$ is mapped onto the incrementally built map to refine the motion estimate $\mathbf{m}_{k-1,B}^{k,B}$. Note that the map here is generated incrementally. This was one of the major contributions of the work by Zhang and Singh (2017) that provided impressive performance locally. The output of the mapping process is $\mathbf{T}_{k,B}^{L}$. However, the mapping process is conducted at a low frequency circle due to computational cost. Therefore, the local transform integration is applied to integrate the high frequency but rough relative motion estimate $(\mathbf{m}_{k-1,B}^{k,B})$ and the low frequency but locally accurate motion estimation $(\mathbf{T}_{k,B}^{L})$. Details of the mapping process can be found in Zhang and Singh (2014).

4.2 | AGPC Detection

This section presents the detection of the AGPC based on the raw 3D point clouds from 3D lidar. For a given epoch *k*, the clouds $\mathbf{P}_k$ usually involve abundant points scanned from the ground surface for ground vehicles installed with 3D lidar. For a typical 3D lidar sensor installed on the roof of a vehicle, the sensor height relative to the ground surface is approximately known. Numerous works (Choi et al., 2014; Yang & Förstner, 2010) were done to extract the ground plane based on the raw 3D point clouds. The *Random Sample Consensus* (RANSAC) is one of the most efficient algorithms for detecting the ground plane parameters from the raw 3D point clouds. In this paper, we define the model parameters of a ground plane as: a) the Euclidean distance from the center of 3D lidar to the detected ground surface; and b) the norm vector of the surface as follows:

$$\mathbf{G}_k = [G_k^x \quad G_k^y \quad G_k^z \quad d_k]^T \tag{4}$$

where the (G_k^x, G_k^y, G_k^z) represents the norm vector and d_k denotes Euclidean distance. Therefore, the ground plane detection problem using the RANSAC can be defined as follows:

$$\mathbf{G}_k^* = \text{argmin} \sum_{\mathbf{P}_{k,i} \in \mathcal{G}_k} f(\mathbf{G}_k, \mathbf{P}_{k,i}) \tag{5}$$

where $\mathbf{G}_k^*$ denotes the optimal parameters of the ground plane. The variable $\mathcal{G}_k$ denotes a set of the given points on the ground surface. The operator $f(*)$ is employed to evaluate the distance between a given point $\mathbf{P}_{k,i}$ and the plane $\mathbf{G}_k$. Given a point $\mathbf{P}_{k,i} = \{x_{k,i}, y_{k,i}, z_{k,i}\}$ and ground plane $\mathbf{G}_k$, the distance can be calculated as follows:

$$f(\mathbf{G}_k, \mathbf{P}_{k,i}) = \left\| G_k^x x_{k,i} + G_k^y y_{k,i} + G_k^z z_{k,i} - d_k \right\| \tag{6}$$

where $f(\mathbf{G}_k, \mathbf{P}_{k,i})$ represents the Euclidean distance. The detail of the plane detection is given in the following Algorithm 1. Note that $\mathbf{P}_k$ is filtered before being input to the algorithm by only keeping the points with z-axis values ranging from −2.5 to 2.5 meters. The parameters required in Algorithm 1 include the minimum number of points $(\mathbf{t}_{np})$ required to detect the ground plane, the maximum number of iterations $(\mathbf{t}_{iter})$ allowed in the RANSAC, a threshold value $(\mathbf{t}_{dis})$ of distance for judging whether a point fits the ground plane, and a threshold of the number of points $(\mathbf{t}_a)$ belonging to the close data indicating that the estimated ground plane model fits well to the points.

The ground detection results based on Algorithm 1 are illustrated in Figure 3 which shows the ground detection with data collected from typical intersections and roadways. Figure 3(a) shows ground detection results in an intersection case. The white points denote the non-ground points and the red points denote the ground points. For the roadway case in Figure 3(b) with the dynamic vehicles located on both sides, the ground points in the front and back of the ego-vehicle are also effectively detected and annotated with red points.

ALGORITHM 1
Ground Plane Detection Using RANSAC

Input: Point clouds $\mathbf{P}_k$
Output: Ground plane parameters $\mathbf{G}_k^*$
Step 1: Initialize $t \leftarrow 0$, $\mathbf{G}_k \leftarrow [0,0,1,2]$ and $\mathbf{E}_t \leftarrow \infty$.
Step 2: while $t > \mathbf{t}_{iter}$

- **Step 2.1**: Select $\mathbf{T}_{np}$ points randomly from $\mathbf{P}_k$ getting $\mathbf{Q}_k$. Fit the $\mathbf{G}_k$ using the selected points $(\mathbf{Q}_k)$. Initialize an empty set $\mathbf{A}_k$.
- **Step 2.2**: For every point $\mathbf{R}_{k,i}$ inside a set $\mathbf{R}_k = \{\mathbf{P}_k - \mathbf{Q}_k\}$, evaluate the distance $(e_{k,i})$ between the $\mathbf{R}_{k,i}$ and the plane $\mathbf{G}_k$. If $e_{k,i} < \mathbf{t}_{dis}$, add $\mathbf{R}_{k,i}$ to $\mathbf{A}_k$.
- **Step 2.3**: If the number of points inside $\mathbf{A}_k$ is larger than $\mathbf{t}_a$, fit the parameters $\mathbf{G}_k$ again based on the $\mathbf{Q}_k$ and $\mathbf{A}_k$. Calculate the fitting error $\mathbf{E}_t$ based on Equation (6). If $\mathbf{E}_t < \mathbf{E}_{t-1}$, assign $\mathbf{G}_k^* = \mathbf{G}_k$.

Step 3: Finish the algorithm and the ground plane parameters are estimated as $\mathbf{G}_k^*$

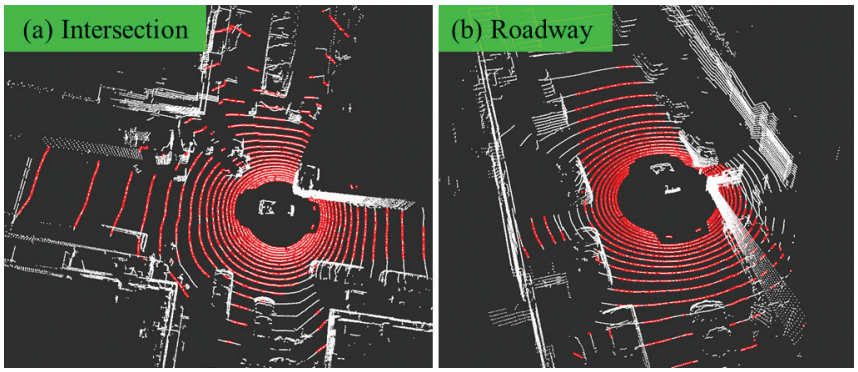

FIGURE 3 Illustration of ground detection using Algorithm 1 (the white points represent the raw point cloud from 3D lidar and the red points denote the detected ground point cloud).

4.3 | Back-End: Loop Closure Detection

Loop closure (Magnusson et al., 2009) is an effective approach for reducing accumulated error over time. The principle of loop closure is to detect the frames of the

point cloud with a large percentage of overlap. The accumulated driving distance is denoted as D_k from the first epoch to the current one. The loop closure detection is performed only when the D_k exceeds a threshold $\mathbf{t}_D$ that is tuned heuristically. Then, the loop closure is checked by matching the current keyframe and historical keyframes using the NDT in Magnusson et al. (2009). Loop closure is found when the fitness condition (Magnusson et al., 2009) is smaller than a given threshold $\mathbf{t}_F$. The output of the loop closure is the relative constraint between the current frame and the historical keyframe as follows:

$$\mathbf{T}_{i,j}^{loop} = (\mathbf{p}_{i,j}^{loop}, \mathbf{q}_{i,j}^{loop})^T \tag{7}$$

where $\mathbf{T}_{i,j}^{loop}$ denotes the transformation between the two keyframes corresponding to the detected loop closure, frame i and j. The $\mathbf{p}_{i,j}^{loop}$ value denotes translation and $\mathbf{q}_{i,j}^{loop}$ represents the rotation in quaternion form.

4.4 | Back-End: Factor Graph-Based Optimization

This section presents the factor graph construction based on the previously derived constraints and optimization. To effectively integrate measurements from lidar odometry (Section 4.1), APGC detection (Section 4.2), and loop closure (Section 4.3), we made use of the state-of-the-art factor graph (Indelman et al., 2012) to formulate sensor fusion as a non-linear optimization problem. Conventional filtering-based sensor fusion such as the extended Kalman filter and its variant (Li et al., 2015) which exploits the first-order Taylor expansion only once prone to get into sub-optimal solutions. The major advantages of the factor graph are the re-linearization and iteration which can enable the optimized states to approach the optimal iteratively. The graph structure of the proposed AGPC-SLAM is shown in Figure 4, which lists three kinds of constraints and ego-vehicle states. Note that the state of the ego-vehicle at epoch k is represented by the $\mathbf{T}_{B,k}^{W}$ based on Equation (1), which encodes the transformation between the body frame and the local world frame.

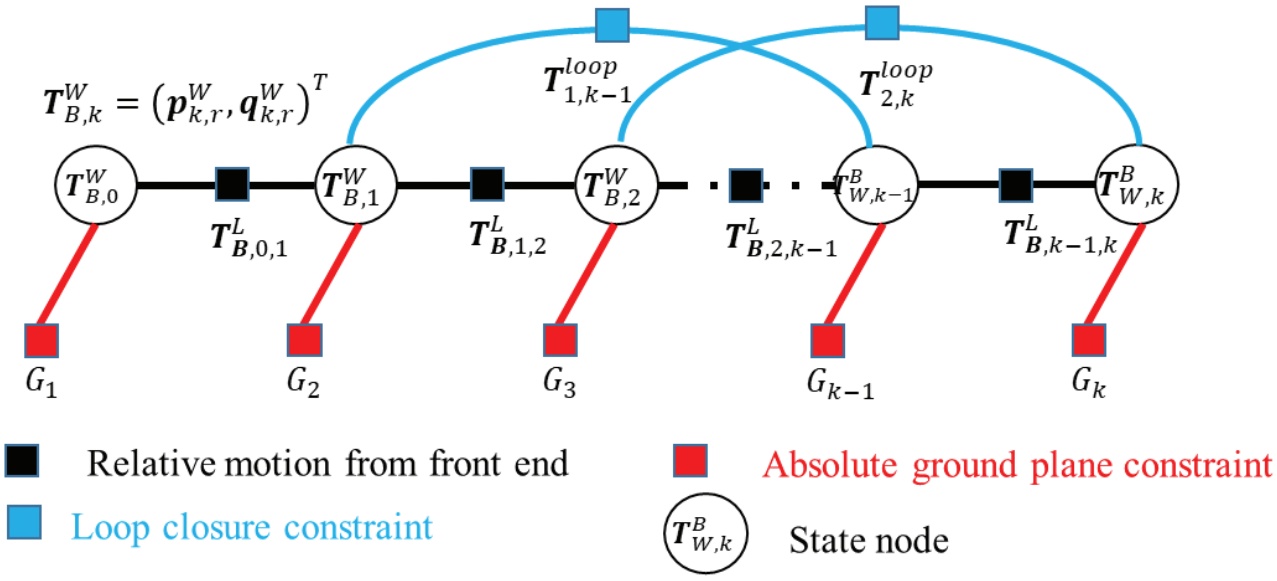

FIGURE 4 Graph structure of the proposed AGPC-SLAM

Following Indelman et al. (2012), the objective of factor graph optimization is to minimize the error function between the observations and the states. The error function for lidar-odometry-derived relative measurements are expressed as:

$$\left\|\mathbf{e}_k^{LiDAR}\right\|^2_{\mathbf{\Sigma}_k^{LiDAR}} = \left\|\left(\mathbf{T}^W_{B,k-1}{}^{-1}\boldsymbol{T}^W_{B,k}\right)\ominus\left(\mathbf{T}^L_{B,k-1}{}^{-1}\boldsymbol{T}^L_{B,k}\right)\right\|^2_{\mathbf{\Sigma}_k^{LiDAR}} \tag{8}$$

where $\mathbf{e}_k^{LiDAR}$ represents the error function for the relative motion between node k and $k+1$ for lidar odometry (front-end). $\mathbf{\Sigma}_k^{LiDAR}$ denotes the information matrix of the error function which is tuned experimentally. $\mathbf{T}^W_{B,k}$ and $\mathbf{T}^W_{B,k-1}$ denote the states at epoch k and $k-1$. $\mathbf{T}^L_{B,k}$ represents the motion which is derived from lidar odometry in Section 4.1. The operator $\ominus$ is the minus operation of the homogeneous matrix in $SE(3)$.

Similarly, the error function for the loop closure between node i and j is as follows:

$$\left\|\mathbf{e}_{i,j}^{loop}\right\|^2_{\mathbf{\Sigma}_{i,j}^{loop}} = \left\|\left(\mathbf{T}^W_{B,i}{}^{-1}\mathbf{T}^W_{B,j}\right)\ominus\mathbf{T}^W_{L,i}\mathbf{T}^{loop}_{i,j}\right\|^2_{\mathbf{\Sigma}_{i,j}^{loop}} \tag{9}$$

where the $\mathbf{\Sigma}_{i,j}^{loop}$ denotes the information matrix for the loop closure constraint $\mathbf{T}_{i,j}^{loop}$ between nodes i and j.

Regarding constraint from the ground plane detection, the measurement contains the distance (d_k) from the center of lidar to the detected ground surface and the norm vector (G_k^x, G_k^y, G_k^z) of the ground surface. d_k does not suffer from drift over time as it is relative to the absolute ground plane. Note that this is satisfied under the assumption that the vehicle is driving on almost plane ground.

Given the pose estimation $\mathbf{T}^W_{B,k}$ at a given epoch k, the $\mathbf{G}_k$ can be transformed into the local world frame as $\mathbf{G}_k^W$:

$$\mathbf{G}_k^W = [G_k^{w,x'} \quad G_k^{w,y'} \quad G_k^{w,z'} \quad d_k{}']^T \tag{10}$$

where the following are satisfied:

$$[G_k^{w,x'} \quad G_k^{w,y'} \quad G_k^{w,z'}]^T = \mathbf{q}_k^W[G_k^x \quad G_k^y \quad G_k^z]^T \tag{11}$$

$$d_k{}' = d_k - \mathbf{p}_k^{W\,T}[G_k^{w,x'} \quad G_k^{w,y'} \quad G_k^{w,z'}]^T \tag{12}$$

After transforming $\mathbf{G}_k$ into the local world frame, this paper applies the minimum parameterization proposed in Ma et al. (2016) $\tau(\mathbf{G}_k^W) = (\theta, \varphi, d)$, where θ, φ, d are the azimuth angle, elevation angle, and distance with respect to $\mathbf{G}_k^W$, respectively. $\tau(\mathbf{G}_k^W)$ can be derived as follows (Koide et al., 2019; Ma et al., 2016):

$$\tau(\mathbf{G}_k^W) = \left[\arctan\left(\frac{G_k^{y'}}{G_k^{x'}}\right) \quad \arctan\left(\frac{G_k^{z'}}{\sqrt{(G_k^{x'2} + G_k^{y'2} + G_k^{z'2})}}\right) \quad d_k{}'\right]^T \tag{13}$$

Therefore, the error function for the ground plane constraint can be expressed as follows:

$$\left\|\mathbf{e}_k^{ground}\right\|^2_{\mathbf{\Sigma}_k^{ground}} = \left\|\tau(\mathbf{G}_k^W) - \tau(\mathbf{G}_0^W)\right\|^2_{\mathbf{\Sigma}_k^{ground}} \tag{14}$$

where the $\mathbf{\Sigma}_k^{ground}$ is the information matrix of the $\tau(\mathbf{G}_k^W)$ which is experimentally determined. $\mathbf{G}_0^W = [0 \quad 0 \quad 1 \quad 0]^T$ is the expected prior concerning the ground constraint.

In this case, we formulate three kinds of error functions for constraints. Therefore, the optimal state set $\mathbf{T}_B^W = \{\mathbf{T}_{B,0}^W, \mathbf{T}_{B,1}^W, \mathbf{T}_{B,2}^W, \ldots, \mathbf{T}_{B,k}^W, \ldots\}$ can be solved as follows:

$$\mathbf{T}_B^{W^*} = \text{argmin} \sum_{k=0,..K} \left(\left\| \mathbf{e}_k^{LiDAR} \right\|^2_{\mathbf{\Sigma}_k^{LiDAR}} + \left\| \mathbf{e}_{i,j}^{loop} \right\|^2_{\mathbf{\Sigma}_{i,j}^{loop}} + \left\| \mathbf{e}_k^{ground} \right\|^2_{\mathbf{\Sigma}_k^{ground}} \right) \tag{15}$$

where $\mathbf{T}_B^{W^*}$ denotes the state set to be estimated. The variable K denotes the number of epochs involved in the optimization. The operation $\sum_{k=0,..K}*$ denotes the summation of all the error functions. The $\mathbf{\Sigma}_k^{LiDAR}$, $\mathbf{\Sigma}_{i,j}^{loop}$, and $\mathbf{\Sigma}_k^{ground}$ denote the information matrix of the three listed types of constraint.

4.5 | Global Mapping and Global Transformation Integration

The point clouds map relative to the local world frame can be obtained by registering all the frames of point clouds based on the state's set $\mathbf{T}_B^W$ optimized in Section 4.4.

To guarantee real-time performance, factor graph optimization was conducted at a frequency of 1 Hz. Each time the optimization finished, the transformation between the local world frame and the local base frame could be derived as follows:

$$\mathbf{T}_{L,k}^W = \mathbf{T}_{B,k}^W (\mathbf{T}_{B,k}^L)^{-1} \tag{16}$$

Note that $\mathbf{T}_{B,k}^L$ was obtained by tracking the relative motion from lidar odometry at a frequency of 10 Hz (the frequency of the raw 3D point clouds). Once $\mathbf{T}_B^L$ is obtained at epoch l and $l > k$, $\mathbf{T}_{B,l}^W$ is derived as follows:

$$\mathbf{T}_{B,l}^W = \mathbf{T}_{L,k}^W \mathbf{T}_{B,l}^L \tag{17}$$

Therefore, $\mathbf{T}_B^W$ can be obtained at a frequency of 10 Hz which is significant for the application with real-time performance requirements.

5 | EXPERIMENT RESULTS AND DISCUSSIONS

To validate the performance of the proposed AGPC-SLAM, we conducted two real experiments in Hong Kong. The first location was Nathan Road, which is an example of one of the typical scenes in Hong Kong. The road was almost flat throughout the test and the driving distance was about 4.2 kilometers. We were fully aware that the proposed method relied on the assumption that the ground was almost flat during operation. Therefore, we conducted the other experiment validation in another location (where the driving distance was about 2.1 kilometers with a partial ramp road) to further evaluate how the proposed method could perform.

During the experiment, 3D lidar (Velodyne 32) was used to collect 3D point clouds. Besides, the NovAtel SPAN-CPT, a GNSS (GPS, GLONASS, and Beidou) RTK/INS (fiber-optic gyroscopes) integrated navigation system was used to provide the ground truth of positioning. The gyro bias in-run stability of the FOG was 1 degree per hour and its random walk was 0.067 degrees per hour. The baseline between the rover and the GNSS base station was within 7 km.

All the data were collected and synchronized using a robotic operation system (ROS; Quigley et al., 2009). All the data were post-processed using a desktop (Intel Core i7-9700K CPU, 3.60Ghz) computer. Please note that the performance of the

TABLE 1
Parameters Used During Experimental Validation

Para.	t_{np}	t_{dis}	t_a	t_{iter}	t_D
Values	800	0.25	500	1000	20.0
Para.	t_F	$\mathbf{\Sigma}_k^{LiDAR}$	$\mathbf{\Sigma}_{i,j}^{loop}$	$\mathbf{\Sigma}_k^{ground}$	t_c
Values	0.80	$10\,\mathbf{I}_{6\times6}$	$10\,\mathbf{I}_{6\times6}$	$4\,\mathbf{I}_{3\times3}$	0.1

proposed method was evaluated by aligning the pose from SLAM and the NovAtel SPAN-CPT to the east, north, and up (ENU) coordinates. The extrinsic transformation matrix between the ENU frame and the local world frame was provided by the NovAtel SPAN-CPT. The parameters used in the proposed AGPC-SLAM are given in Table 1.

5.1 | Experimental Validation in Location 1

The experimental setup is shown in Figure 5. During the evaluation, we compared the proposed AGPC-SLAM framework with the state-of-the-art LeGO-LOAM (Shan & Englot, 2018).

5.1.1 | *Evaluation Metrics*

As Figure 5(b) shows, the fixed solution was hard to obtain using NovAtel SPAN-CPT throughout the evaluated Location 1. This was caused by low satellite

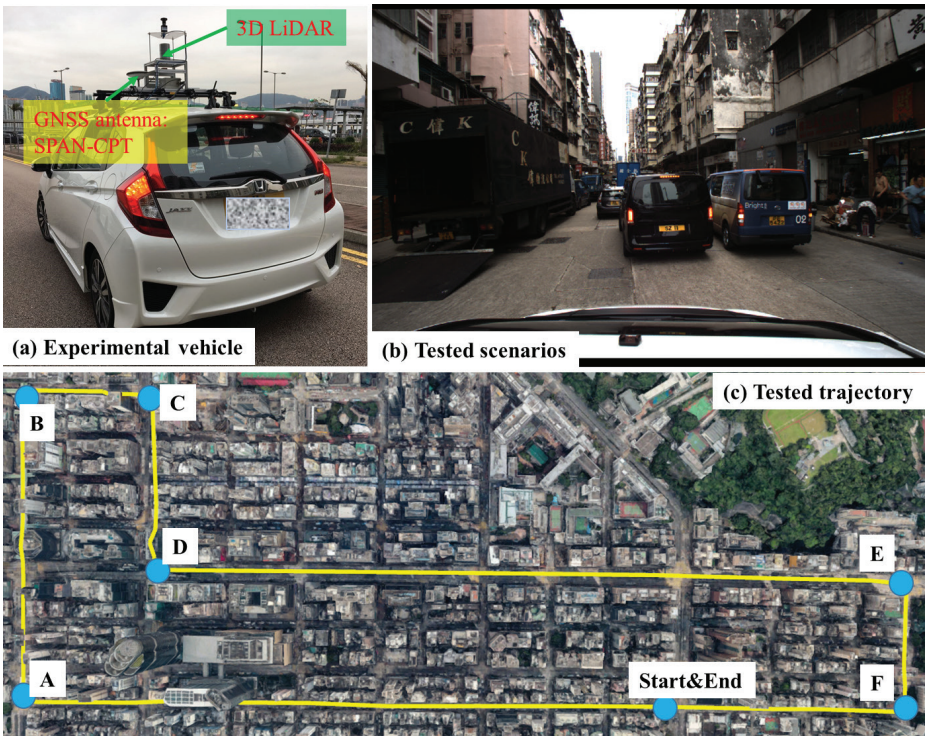

FIGURE 5 (a) Data collection vehicle with all the sensors installed in a compact sensor kit; (b) tested scenarios; and (c) trajectory with a driving distance of 4.2 km

visibility due to the dense and tall buildings. However, we found that the positioning status of the NovAtel SPAN-CPT was healthy in several road intersections (see A, B, C, D, E, and F in Figure 5[c]) where the satellite geometry was better. We carefully checked the positioning results of the NovAtel SPAN-CPT in the selected six intersection points and at the very least, the float solution was obtained. Therefore, we propose to evaluate the performance of the proposed method based on the following metrics.

- **Ground control point (GCP) error:** Evaluating the accuracy at six selected GCPs (A–F in Figure 5); both absolute translation and rotation errors were evaluated.
- **Accumulated error**: Evaluating the accumulated error after driving a loop with/without the loop closure constraints; without loss of generality, the accumulated error metric was also adopted in the KITTI data set (Geiger et al., 2012) for performance evaluation of lidar odometry.

5.1.2 | Performance Analysis

The mean errors of the rotation and translation of the six selected GCPs used are shown in Tables 2 and 3, respectively. Regarding the mean rotation error (Table 2), the LeGO-LOAM introduced a mean error in pitch angle (6.93°). The error in yaw and roll were 2.57° and 2.16°, respectively. With the help of the proposed AGPC, the error in pitch angle decreased to 1.35° which shows that the proposed method can effectively mitigate the drift in pitch angle. The performance in the yaw and roll angle also improved slightly.

Regarding the mean translation error, the errors of LeGO-LOAM in the east and north direction were 3.89 and 11.58 meters, respectively. Meanwhile, the error in

TABLE 2
Performance of Rotation Estimation at Location 1

Method	Roll	Pitch	Yaw	Total	%
LeGO-LOAM (Shan & Englot, 2018)	2.57°	6.93°	2.16°	7.70°	0.18%
AGPC-SLAM (proposed)	1.36°	1.35°	2.61°	3.24°	0.07%
AGPC-SLAM (with loop closure)	1.29°	1.21°	1.53°	2.33°	0.056%

Note: %: [Total]/[total driving distance]

TABLE 3
Performance of the Translation Estimation at Location 1

Method	East	North	Up	ENU	AE	%
LeGO-LOAM (Shan & Englot, 2018)	3.89m	11.58m	43.83m	45.49m	34.80m	1.08%
AGPC-SLAM	1.39m	2.65m	0.41m	3.02m	1.89m	0.04%
AGPC-SLAM (loop closure)	1.32m	2.36m	0.40m	2.73m	0.52m	0.065%

Note: AE: accumulated absolute error. %: [ENU]/[total driving distance]

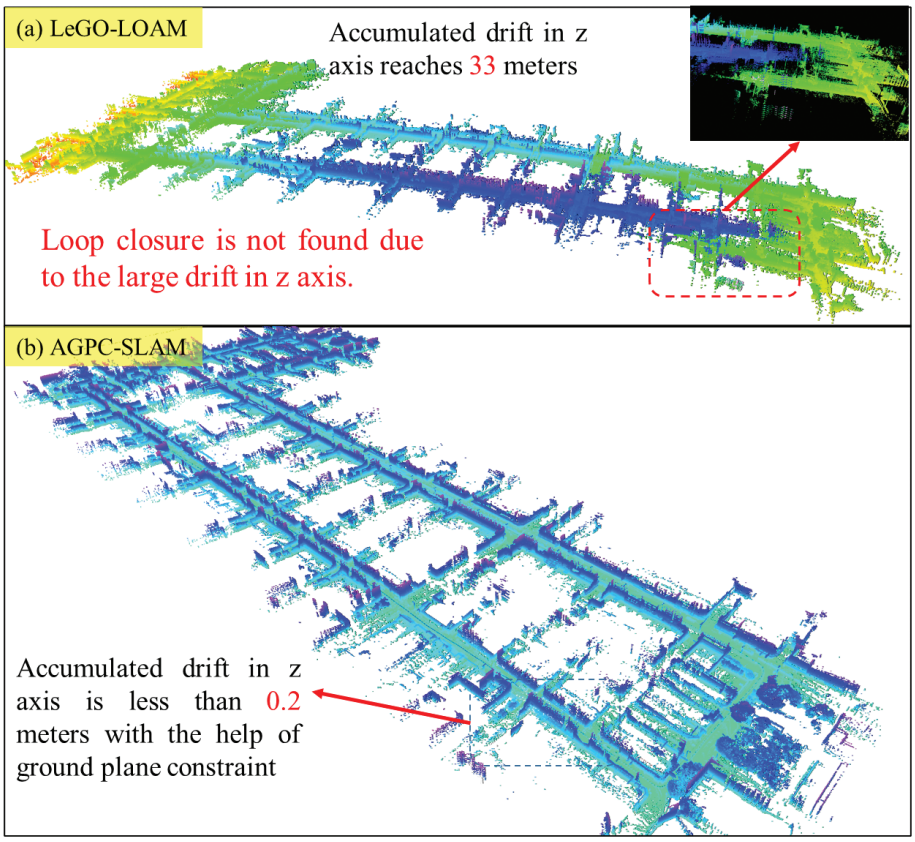

FIGURE 6 Illustration of the map generated by the two methods; the color of the map is annotated by the value of the z-axis (height) of each point.

the altitude (up) was about 43.83 meters. The total mean error in east, north, and up directions was 45.49 meters. The accumulated error (the error at the last epoch) reached 34.80 meters. The last column represents the percentage of mean error concerning the total driving distance. With the help of the proposed AGPC-SLAM, the mean error decreased to 3.02 meters and the accumulated error was only 1.89 meters. A significant improvement in the vertical direction shows that the proposed AGPC can effectively constrain the drift. Meanwhile, the detailed performances at six GCPs can be found in the Appendix of this paper (see Figure 12 and Figure 13).

After applying the loop closure constraint, the LeGO-LOAM could not successfully detect the loop closure due to the large drift which can be seen in the top panel of Figure 6. Therefore, the loop closure result for LeGO-LOAM is not provided in Tables 2 and 3. However, the rotation estimation for the proposed method improved with the help of loop closure with a mean rotation error of 2.33° and the result of the map can be seen in the bottom panel Figure 6. With the help of the loop closure, the mean error of translation decreased to 2.73 meters and the accumulated error was only 0.52 meters. The percentage of the error was only 0.065%. Although the loop closure detection is not a major contribution of this paper, the results show that the proposed AGPC could further enhance the detection of the loop closure.

Figure 7 shows the details of the generated point map with the color annotated by the reflectivity. The lane lines and building boundaries are clear which can be seen from Figure 7(a) and Figure 7(b). A detailed video of the proposed

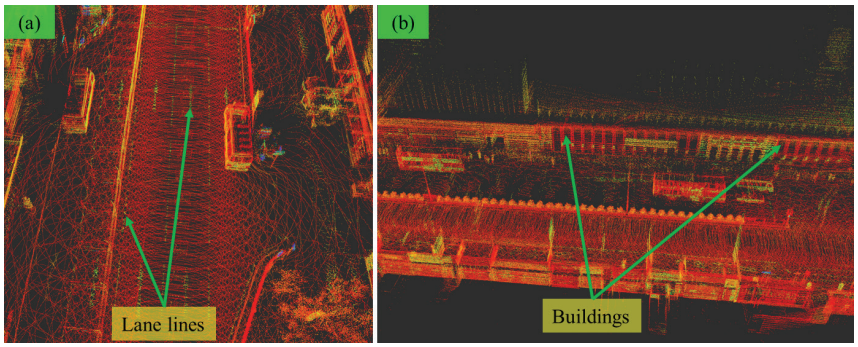

FIGURE 7 Details of the generated map with reflectivity using the proposed AGPC-SLAM in Location 1

AGPC-SLAM in Location 1 can be found at the following link: https://www.youtube.com/watch?v=3nS895StJUo&feature=youtu.be.

5.2 | Experimental Validation in Location 2

5.2.1 | Evaluation Metrics

To further evaluate how the proposed method could perform in the scenario with a partial ramp road, we performed the other experiment in Tsim Sha Tsui of Hong Kong. The location is shown in Figure 8 where the red dots denote the evaluated trajectory. The ramp road starts with an altitude of 0 meters (the height is relative to the start point), increases to a maximum altitude of 2 meters, and decreases to 0 meters by the end of the ramp road. More importantly, the fixed solutions were available frequently during the test. Therefore, the pose from the NovAtel SPAN-CPT was directly used as the ground truth solution of Location 2.

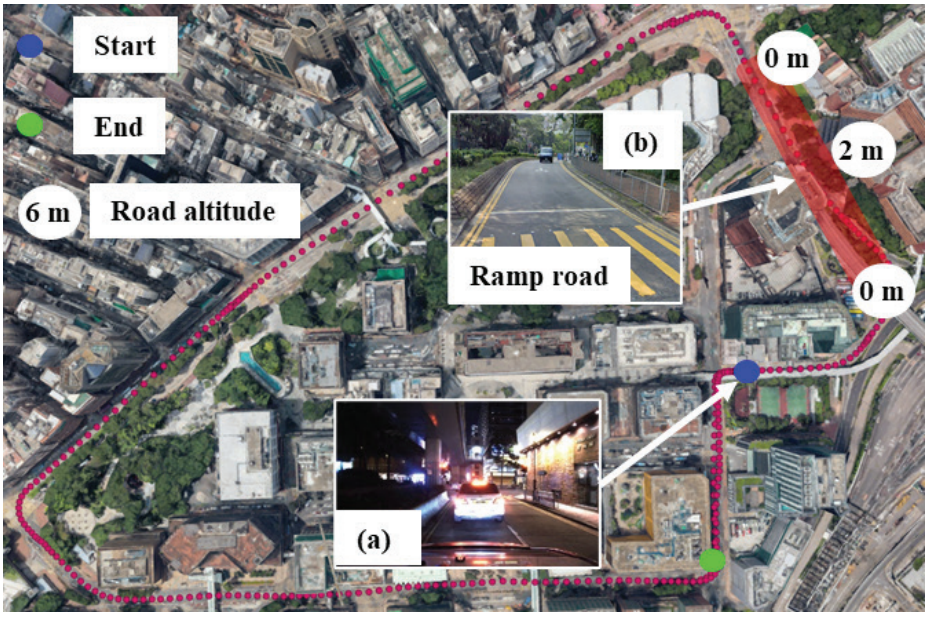

FIGURE 8 (a) Environmental condition at the start point (blue shaded circle); (b) the scene with partial ramp road (red shaded area)

Lidar SLAM only provides the relative positioning concerning the starting point, therefore, we applied the popular evaluating odometry (EVO; Grupp, 2017) toolkit to evaluate the relative error (RPE) for translation and rotation. Meanwhile, the evaluation metrics are as follows:

- **RMSE**: Root-mean-square error of the relative translation and rotation
- **MEAN**: Mean error of the relative translation and rotation

To further show the effectiveness of the proposed method in mitigating the vertical and overall drift, we added three additional evaluation metrics as follows:

- **Altitude**: the accumulated altitude drift by the end of the trajectory
- **AE**: the accumulated error of translation (meters) and rotation (degrees) by the end of the trajectory
- **%**: calculated by the [accumulated drift]/[total driving distance]; this was adopted to evaluate the overall drift of the SLAM algorithm.

5.2.2 | Performance Analysis

Table 4 shows the performance of the translation and rotation estimation using LeGO-LOAM and the proposed AGPC-SALM, respectively. Note that the loop closure is not applied regarding the results in Table 4 to evaluate the contribution of the standalone AGPC. Regarding the translation, both the MEAN and RMSE were reduced with the help of the AGPC using the proposed method. Although the LeGO-LOAM proposed to estimate the motion of the z-axis and pitch angle using ground points, the final altitude drift still reached 7.36 meters by the end of the test. Fortunately, the drift in altitude decreased to only 0.21 meters with the help of the AGPC, which shows the effectiveness of the proposed AGPC. Meanwhile, the overall drift was also reduced using the proposed AGPC-SLAM (see the AE and % columns in Table 4).

The continuous 3D trajectories of LeGO-LOAM, AGPC-SLAM, and ground truth are shown in Figure 9. Specifically, the altitude during the test is shown in Figure 10. The black curve denotes the altitude provided by the NovAtel SPAN-CPT. We can see that the altitude estimation using the LeGO-LOAM deviated significantly from the ground truth during epoch 150~240 which corresponds to the red shaded area

TABLE 4
Performance of the Translation and Rotation Estimation in Location 2

Method	MEAN	RMSE	Altitude	AE	%
LeGO-LOAM (Shan & Englot, 2018) (Translation)	0.33 m	0.46 m	7.36 m	8.92 m	0.42%
AGPC-SLAM (Translation)	0.27 m	0.38 m	0.21 m	5.26 m	0.26%
LeGO-LOAM (Shan & Englot, 2018; Rotation)	0.69°	1.22°		3.72°	0.19%
AGPC-SLAM (Rotation)	0.58°	1.07°		1.87°	0.09%

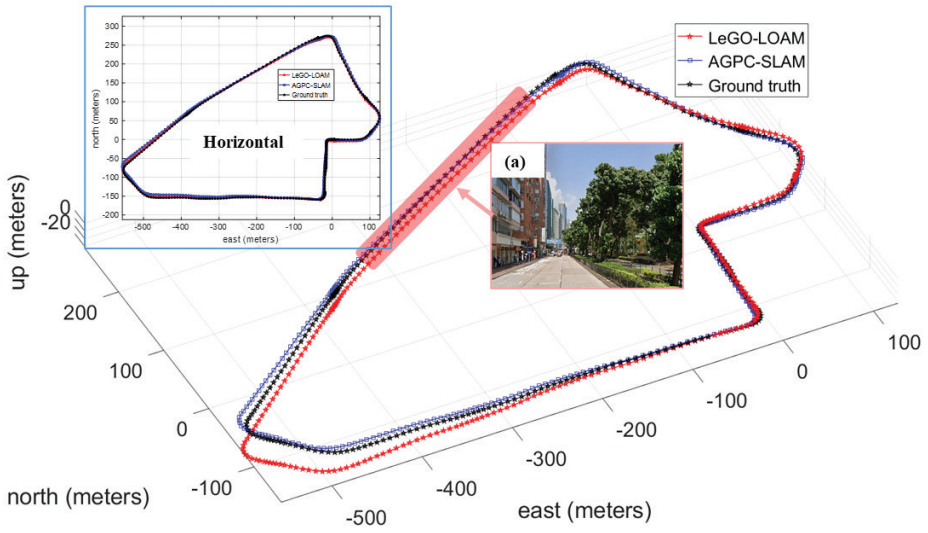

FIGURE 9 Trajectories of the LeGO-LOAM (red curve), AGPC-SLAM (blue curve), and ground truth trajectory (black curve)

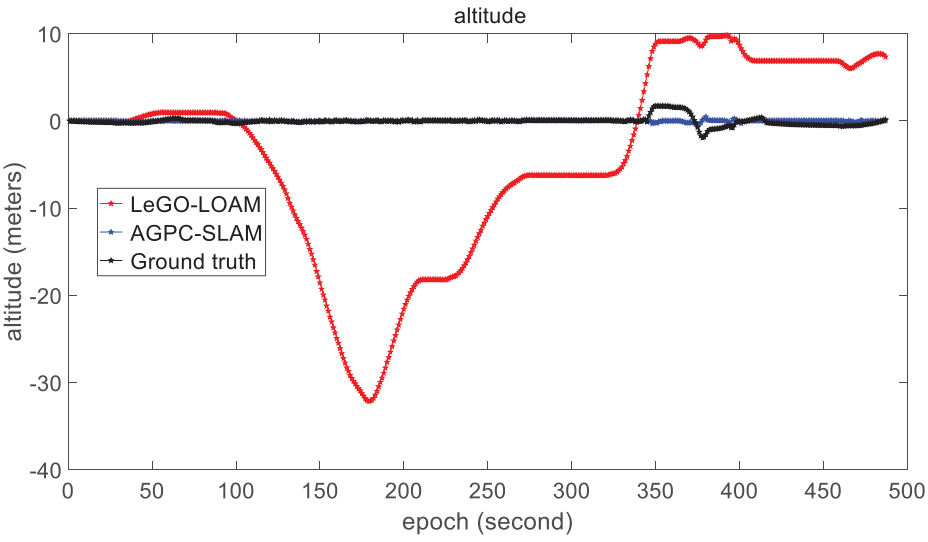

FIGURE 10 Altitude of the LeGO-LOAM (red curve), AGPC-SLAM (blue curve), and ground truth trajectory (black curve) during the testing area with the ramp road

in Figure 9. During this period, one side of the road was filled with building surfaces which caused it to be a feature-insufficient location. The other side had dense foliage with numerous tree leaves that increased the difficulty of finding the correct correspondence of LOAM. As a result, the LeGO-LOAM drift significantly in the altitude direction. With the help of the AGPC, the altitude drift was mitigated accordingly.

The RPE of the translation and rotation estimations are provided in Figure 11. Improvements can be seen in the figure for both translation and rotation. The red and blue dash curves denote the translation errors of LeGO-LOAM and AGPC-SLAM, respectively. The black and cyan dash curves denote the rotation errors of LeGO-LOAM and AGPC-SLAM, respectively.

As shown in Figure 8, Location 2 involved a ramp road with an altitude deviating from 0 to 2 meters, which corresponds to the annotated area by the red rectangle in Figure 9. The estimated altitude using AGPC-SLAM deviates from the ground

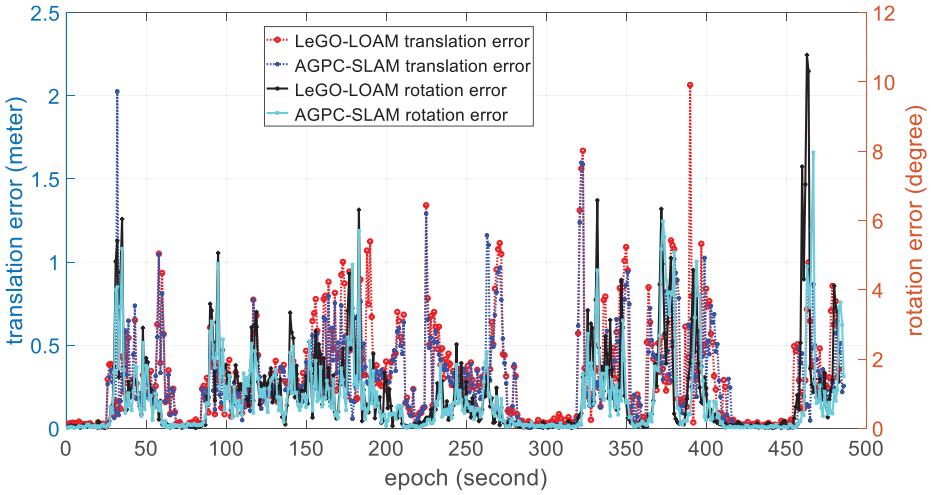

FIGURE 11 Relative translation and rotation errors for LeGO-LOAM and the proposed AGPC-SLAM, respectively

TABLE 5
Performance of Translation and Rotation Estimation with Loop Closure

Method	Trans. RMSE	Rot. RMSE	Altitude
LeGO-LOAM (Shan & Englot, 2018; Loop closure)	Loop Not Detected	Loop Not Detected	Loop Not Detected
AGPC-SLAM (Loop closure)	0.43 m	0.64°	0.13 m

truth altitude (black curve). This was caused by the violation of the assumption that the ground plane was flat during operation, which is required by the proposed method in this paper. In the future, we will study the ramp road surface identification and remove the AGPC when the slope surface is detected. A detailed video of the test in Location 2 can be found at the following link: https://www.youtube.com/watch?v=mgLxxlhlscY&feature=youtu.be.

The performance of the translation and rotation angle estimation of AGPC-SLAM with loop closure is presented in Table 5. Although the loop closure was enabled in LeGO-LOAM, the loop was not successfully detected due to the large drift in the z-axis of LeGO-LOAM. However, the loop closure was detected using the proposed AGPC-SLAM as the vertical drift was improved. We believe that this is another contribution of the AGPC which improves the detection of the loop closure.

6 | CONCLUSION AND FUTURE WORK

To mitigate the drift of lidar SLAM in the vertical direction due to the limited field of view of the lidar sensor, this paper proposes an AGPC-SLAM method that achieved improved accuracy compared to the existing state-of-the-art LeGO-LOAM. This paper innovatively proposes to employ absolute ground plane detection to mitigate the drift in z-axis related states. Moreover, better-constrained positioning in the z-axis and pitch angle can also improve the positioning in the x-axis and y-axis. We tested the proposed method in two typical scenarios in Hong Kong. The results show that the proposed method can effectively mitigate the drift on the z-axis in both the evaluated scenarios.

One of the limitations of our work is that the proposed method is effective only when a flat ground plane is available. In some extreme cases, the ground can be fully occluded by surrounding dynamic vehicles which can lead to misidentification of the ground surface. The ground surface can also be a slope that cannot be simply modeled using a plane function. In the future, we will study the scene with slope and derive the corresponding constraints to alleviate the drift of 3D lidar SLAM in urban canyons.

REFERENCES

Bai, X., Wen, W., & Hsu, L.-T. (2020). Robust visual-inertial integrated navigation system aided by online sensor model adaption for autonomous ground vehicles in urban areas. *Remote Sensing, 12*(10). https://doi.org/10.3390/rs12101686

Chang, L., Niu, X., & Liu, T. (2020). GNSS/IMU/ODO/LiDAR-SLAM integrated navigation system using IMU/ODO pre-integration. *Sensors, 20*(17). https://doi.org/10.3390/s20174702

Choi, S., Park, J., Byun, J., & Yu, W. (2014). Robust ground plane detection from 3D point clouds. *2014 14th International Conference on Control, Automation and Systems (ICCAS 2014*), Gyeonggi-do, South Korea. https://doi.org/10.1109/ICCAS.2014.6987936

Dill, E. T., & Uijt de Haag, M. (2016). 3D multi-copter navigation and mapping using GPS, inertial, and LiDAR. *NAVIGATION, 63*(2), 205–220. https://doi.org/10.1002/navi.134

Dow, J. M., Neilan, R. E., & Rizos, C. (2009). The international GNSS service in a changing landscape of global navigation satellite systems. *Journal of Geodesy, 83*(3), 191–198. http://doi.org/10.1007/s00190-009-0315-4

Geiger, A., Lenz, P., & Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI vision benchmark suite. *2012 IEEE Conference on Computer Vision and Pattern Recognition,* Providence, RI. https://doi.org/10.1109/CVPR.2012.6248074

Grisetti, G., Kummerle, R., Stachniss, C., & Burgard, W. (2010). A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine, 2*(4), 31–43. https://doi.org/10.1109/MITS.2010.939925

Grupp, M. (2017). evo: Python package for the evaluation of odometry and SLAM. https://github.com/MichaelGrupp/evo

He, G., Yuan, X., Zhuang, Y., & Hu, H. (2020). An integrated GNSS/LiDAR-SLAM pose estimation framework for large-scale map building in partially GNSS-denied environments. *IEEE Transactions on Instrumentation and Measurement, 70*. https://doi.org/10.1109/TIM.2020.3024405

Hess, W., Kohler, D., Rapp, H., & Andor, D. (2016). Real-time loop closure in 2D LIDAR SLAM. *2016 IEEE International Conference on Robotics and Automation (ICRA),* Stockholm, Sweden. https://doi.org/10.1109/ICRA.2016.7487258

Huang, X., Wang, P., Cheng, X., Zhou, D., Geng, Q., & Yang, R. (2019). The ApolloScape open dataset for autonomous driving and its application. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 42*(10)*, 2702–2719*. https://doi.org/10.1109/TPAMI.2019.2926463

Indelman, V., Williams, S., Kaess, M., & Dellaert, F. (2012). Factor graph based incremental smoothing in inertial navigation systems. *2012 15th International Conference on Information Fusion,* Singapore. https://ieeexplore.ieee.org/document/6290565

Koide, K., Miura, J., & Menegatti, E. (2019). A portable three-dimensional LIDAR-based system for long-term and wide-area people behavior measurement, *International Journal of Advanced Robotic Systems, 16*(2). https://doi.org/10.1177/1729881419841532

Kuramachi, R., Ohsato, A., Sasaki, Y., & Mizoguchi, H. (2015). G-ICP SLAM: An odometry-free 3D mapping system with robust 6DoF pose estimation. *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Zhuhai, China. https://doi.org/10.1109/ROBIO.2015.7418763

Li, Q., Li, R., Ji, K., & Dai, W. (2015). Kalman filter and its application. *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS),* Tianjin, China. https://doi.org/10.1109/ICINIS.2015.35

Lin, J., & Zhang, F. (2020). Loam livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV. *2020 IEEE International Conference on Robotics and Automation (ICRA),* Paris, France. https://doi.org/10.1109/ICRA40945.2020.9197440

Low, K.-L. (2004). *Linear least-squares optimization for point-to-plane ICP surface registration* (Technical Report TR04-004). University of North Carolina at Chapel Hill. https://www.comp.nus.edu.sg/~lowkl/publications/lowk_point-to-plane_icp_techrep.pdf

Ma, L., Kerl, C., Stückler, J., & Cremers, D. (2016). CPA-SLAM: Consistent plane-model alignment for direct RGB-D SLAM. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden. https://doi.org/10.1109/ICRA.2016.7487260

Magnusson, M., Andreasson, H., Nuchter, A., & Lilienthal, A. J. (2009). Appearance-based loop detection from 3D laser data using the normal distributions transform. *2009 IEEE International Conference on Robotics and Automation,* Kobe, Japan. https://doi.org/10.1109/ROBOT.2009.5152712

Magnusson, M., Lilienthal, A., & Duckett, T. (2007). Scan registration for autonomous mining vehicles using 3D-NDT. *Journal of Field Robotics: Special Issue on Mining Robotics, 24*(10), 803–827. https://doi.org/10.1002/rob.20204

Mascaro, R., Teixeira, L., Hinzmann, T., Siegwart, R., & Chli, M. (2018). GOMSF: Graph-optimization based multi-sensor fusion for robust UAV pose estimation. *2018 IEEE International Conference on Robotics and Automation (ICRA*), Brisbane, Australia. https://doi.org/10.1109/ICRA.2018.8460193

Pang, S., Kent, D., Cai, X., Al-Qassab, H., Morris, D., & Radha, H. (2019). 3D scan registration based localization for autonomous vehicles—A comparison of NDT and ICP under realistic conditions. *2018 IEEE 88th Vehicular Technology Conference,* Chicago, IL. https://doi.org/10.1109/VTCFall.2018.8690819

Qin, T., Li, P., & Shen, S. (2018). VINS-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics, 34*(4), 1004–1020. https://doi.org/10.1109/TRO.2018.2853729

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., & Ng, A. (2009). ROS: An open-source Robot Operating System. *ICRA Workshop on Open Source Software.* http://www.cim.mcgill.ca/~dudek/417/Papers/quigley-icra2009-ros.pdf

Saarinen, J., Andreasson, H., Stoyanov, T., & Lilienthal, A. J. (2013). Normal distributions transform Monte-Carlo localization (NDT-MCL). *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems,* Tokyo, Japan. https://doi.org/10.1109/IROS.2013.6696380

Shan, T., & Englot, B. (2018). LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain. https://doi.org/ 10.1109/IROS.2018.8594299

Shan, T., Englot, B., Meyers, D., Wang, W., Ratti, C., & Rus, D. (2020). LIO-SAM: Tightly-coupled lidar inertial odometry via smoothing and mapping. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* Las Vegas, NV. https://doi.org/10.1109/IROS45743.2020.9341176

Shetty, A., & Gao, G. X. (2019). Adaptive covariance estimation of LiDAR-based positioning errors for UAVs. *NAVIGATION, 66*(2), 463–476. https://doi.org/10.1002/navi.307

Wen, W., Bai, X., Kan, Y. C., & Hsu, L.-T. (2019a). Tightly coupled GNSS/INS integration via factor graph and aided by fish-eye camera. *IEEE Transactions on Vehicular Technology, 68*(11), 10651–10662. https://doi.org/10.1109/TVT.2019.2944680

Wen, W., Hsu, L.-T., & Zhang, G. (2018). Performance analysis of NDT-based graph SLAM for autonomous vehicle in diverse typical driving scenarios of Hong Kong. *Sensors, 18*(11). https://doi.org/10.3390/s18113928

Wen, W., Zhang, G., & Hsu, L.-T. (2020). Object-detection-aided GNSS and its integration with lidar in highly urbanized areas. *IEEE Intelligent Transportation Systems Magazine, 12*(3), 53–69. https://doi.org/10.1109/MITS.2020.2994131

Wen, W., Zhang, G., & Hsu, L.-T. (2019b). Correcting NLOS by 3D LiDAR and building height to improve GNSS single point positioning. *NAVIGATION, 66*(4), 705–718. https://doi.org/10.1002/navi.335

Yang, M. Y., & Förstner, W. (2010). *Plane detection in point cloud data* (Technical Report No. 1). Institute of Geodesy and Geoinformation at the University of Bonn. http://www.ipb.uni-bonn.de/pdfs/Yang2010Plane.pdf

Ye, H., Chen, Y., & Liu, M. (2019). Tightly coupled 3D lidar inertial odometry and mapping. *2019 International Conference on Robotics and Automation (ICRA),* Montreal, Canada. https://doi.org/10.1109/ICRA.2019.8793511

Zhang, J., & Singh, S. (2014). LOAM: Lidar odometry and mapping in real-time. *Robotics: Science and Systems Conference*, Berkley, CA. https://doi.org/10.15607/RSS.2014.X.007

Zhang, J., & Singh, S. (2017). Low-drift and real-time lidar odometry and mapping. *Autonomous Robots, 41*(2), 401–416. https://doi.org/10.1007/s10514-016-9548-2

Zhao, S., Fang, Z., Li, H., & Scherer, S. (2019). A robust laser-inertial odometry and mapping method for large-scale highway environments. *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* Macau, China. https://doi.org/10.1109/IROS40897.2019.8967880

Zheng, L., Zhu, Y., Xue, B., Liu, M., & Fan, R. (2019). Low-cost GPS-aided lidar state estimation and map building. *2019 IEEE International Conference on Imaging Systems and Techniques (IST),* Abu Dhabi, United Arab Emirates. https://doi.org/10.1109/IST48021.2019.9010530

Zuo, X., Geneva, P., Lee, W., Liu, Y., & Huang, G. (2019). LIC-Fusion: LiDAR-inertial-camera odometry. *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* Macau, China. https://doi.org/10.1109/IROS40897.2019.8967746

How to cite this article: Hsu, L.-T., & Wen, W. (2022). AGPC-SLAM: Absolute ground plane constrained 3D lidar SLAM. *NAVIGATION, 69*(3). https://doi.org/10.33012/navi.527

APPENDIX: PERFORMANCE OF AGPC-SLAM AT 6 GCPS

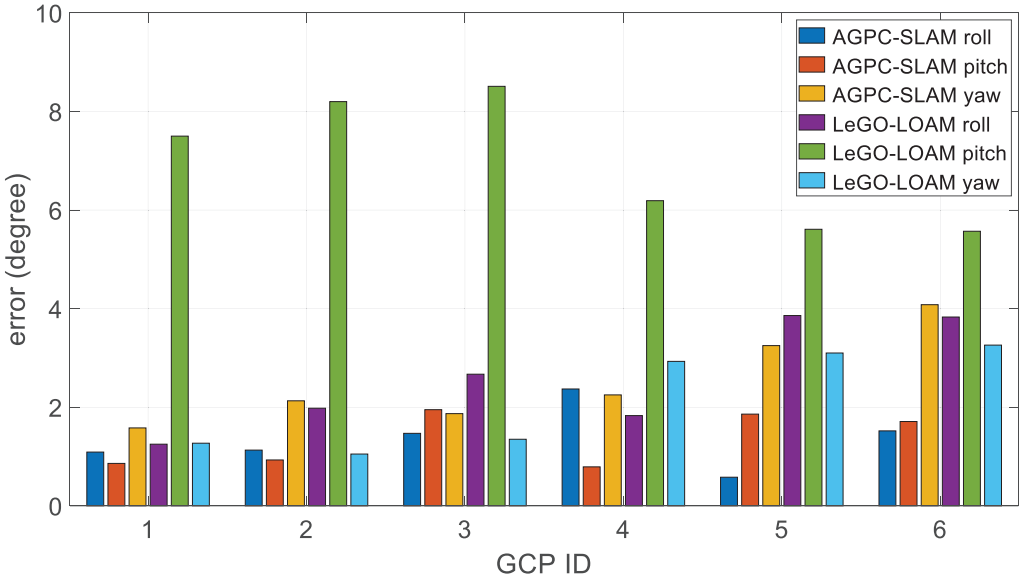

FIGURE 12 Errors of the roll, pitch, and yaw angles for LeGO-LOAM and the proposed AGPC-SLAM (the x-axis denotes the ID of the GCPs from 1 to 6 and the y-axis denotes the errors)

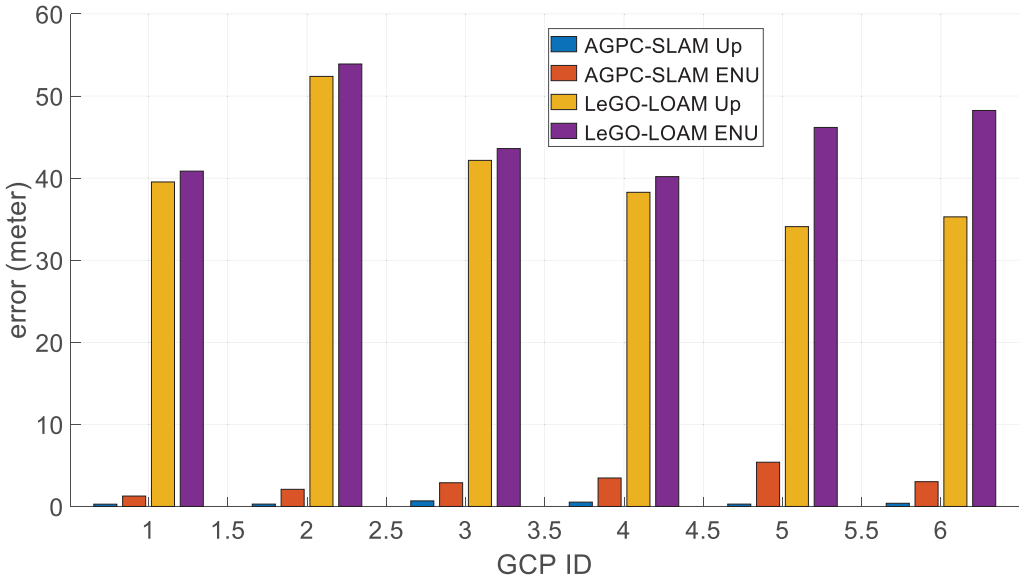

FIGURE 13 Errors of the up (altitude) and ENU for LeGO-LOAM and the proposed AGPC-SLAM, respectively (the x-axis denotes the ID of the GCPs from 1 to 6 and the y-axis denotes the errors)