Received: 5 February 2020 | Revised: 14 September 2020 | Accepted: 15 October 2020

DOI: 10.1002/navi.403

SPECIAL ISSUE ARTICLE

Multi-slices navigation approach for unknown 3D environments using micro aerial vehicles

H. A. Mohamed[1] | A. M. Moussa[1,2] | N. El-Sheimy[1] | M. M. Elhabiby[3]

[1] Department of Geomatics Engineering, University of Calgary, Calgary, Alberta, Canada

[2] Department of Electrical and Computer Engineering, Port-Said University, Egypt

[3] Public Works Department, Ain Shams University, Cairo

Correspondence
H. A. Mohamed, Department of Geomatics Engineering, University of Calgary, Calgary, Alberta, Canada.
Email: haytham.abdalla@ucalgary.ca

Abstract
Due to the small size of Micro Aerial Vehicles, they are well suited for various indoor applications, especially search/rescue operations. Most of these operations are performed in unknown 3D environments. Real-time map construction and vehicle's localization are essential tasks. Various approaches provide solutions for 3D-map representation. However, these approaches require expensive embedded systems to afford high-processing memory/computational costs. Because of its exposure to risks, the MAV should be equipped with a low-cost navigation system. The principal aim of map construction herein is to facilitate the navigation task. Thus, constructing a massive 3D map is not required. Consequently, this paper proposes an efficient real-time 3D SLAM. The proposed method avoids the 3D-map representation of each region of the trajectory. Alternatively, it divides the environment along the trajectory into several 2D maps with a single 2D map in every region at the height of the MAV, as well as a transient region between each of the two constructed maps to enable connecting neighboring maps.

KEYWORDS
3D Reconstruction, Robust Navigation in GNSS-Denied and GNSS Challenged Environments, Indoor Navigation, Algorithms and Methods, Autonomous Navigation, Autonomous Robot and Vehicle Navigation

1 | INTRODUCTION

Over the past few decades, the potential use of the Micro Aerial Vehicles (MAVs) in search and rescue operations increased because they can be in action promptly without the loss of time. In addition to the quick preparation, their small size grants them the preference against their peers to efficiently explore narrow environments. Typically, the MAVs encounter unknown 3D environments without erstwhile information for most search and rescue operations. Therefore, to successfully achieve the navigation task in such environments, the MAV must construct a map for the environment and concurrently locate itself within this map using the onboard sensors. The simultaneous localization and mapping (SLAM) approach is utilized to accomplish such a problem (Alpen et al. 2012; Kamarudin et al., 2015; Kohlbrecher et al. 2011). However, constructing a massive 3D map is not required in search and rescue operations. The vehicle utilizes the real-time map construction to accomplish the navigation task.

Hence, for such operations, estimating accurate navigation parameters of unmanned vehicles is the main objective of the SLAM approach (Thrun, 2007). Furthermore, to successfully complement the navigation, the vehicle has to perform a collision-free trajectory to safely fly from one position to another (Gageik et al., 2015; Kalogeiton et al., 2019).

Typically, constructing maps for 3D environments often suffers from high memory and computational costs due to the huge amount of collected data (Khan et al., 2015; Triebel et al., 2006). On one hand, expensive embedded systems with high specifications are employed to handle the processing time of the computations and the required memory. On the other hand, many accustomed approaches propose compact representation to reduce the required memory and computation load. Thus, the compact representation is much more proper for search and rescue operations because the utilized MAVs are susceptible to loss during harsh situations. For efficient 3D mapping, many probabilistic approaches such as the voxel occupancy grids (Duffy et al., 1989; Plaza-Leiva et al., 2015), elevation map (Hadsell et al., 2009; Herbert et al., 1989), multi-level surface maps (MLS) (Rivadeneyra et al., 2009; Triebel et al., 2006), Octomap (Hornung et al., 2013; Wurm et al., 2010), multi-volume occupancy grid (MVOG) (Dryanovski et al., 2010), multi-level occupancy grid (MLOG) (Tian et al., 2016), and Occupancy Elevation Grid (OEG) (Souza & Gonçalves, 2016) are used.

The voxel occupancy grids approach is represented as the direct extension of the 2D occupancy grid for the 3D environment (Anderson-Sprecher & Simmons, 2012). This approach represents the 3D environment as a grid of cubic volumes (i.e., voxel) of equal size, where each cube of the grid is independent from its neighbors. However, this approach suffers from large memory requirements as the whole environment must be represented by voxels. Although the small voxel represents a close representation to the real environment, it suffers from the high memory consumption and high processing cost. In order to reduce the memory consumption, the voxel size is enlarged, which leads to a coarser representation of the environment. The octomap or octree approach is a hierarchical data structure in which each parent voxel has exactly eight children (Jessup et al., 2014; Zhang et al., 2018). Thus, the 3D environment is represented as a collection of voxels with varied sizes, and hence, some voxels are recursively subdivided into eight children until a desired resolution is achieved. Although this approach postpones the initialization of the grid size until measurements need to be updated and it reduces the number of the required voxels, it also suffers from large memory requirements.

The elevation map approach represents the 3D environment in a compact representation using the 2D occupancy grid. Each cell in the grid stores the height of the obstacle in the corresponding area (Pfaff & Burgard, 2006). However, this approach suffers from the insufficient representation of the vertical structures or even the multi-level surfaces. The multi-level surface maps (MLS) approach is similar to the elevation map approach except for allowing the storage of different heights in the same cell to clearly represent the vertical structures (Pfaff, Triebel, & Burgard, 2007). Hence, this approach consists of a horizontal 2D grid structure, where each cell stores different elevation values in the corresponding area. Although the multi-level surface maps approach offers compact representation, it only records positive sensor data, and it does not provide a mechanism for changing the recorded occupancy value of the object's elevation. Therefore, sensor errors are never removed from the grid.

The multi-volume occupancy grid (MVOG) approach is an extension of the MLS approach (Morris et al., 2010). However, the MVOG approach stores both the obstacles and free space readings, which is denoted as positive and negative readings, respectively. The drawback of this approach is the large memory consumption. The multi-level occupancy grids (MLOG) approach represents the entire 3D environment as a stack of multiple 2D occupancy grids (Tian et al., 2016). Each 2D occupancy grid has a specific height. The height difference between each two 2D occupancy grid is adjusted according to the application requirements. This approach also has the same trait, which is the large memory requirements.

This paper proposes an efficient real-time 3D SLAM method using multiple 2D point cloud slices. This method depends on low-cost sensors to fit the tough situations of the search and rescue operations. The method begins by leveling the 2D point cloud of the laser scanner rangefinder using the inertial sensor as a phase I. Since MAVs are able to translate and rotate around all its axes (6-DOF), this phase aims to mitigate the impact of the rotations (i.e., roll and pitch) of the vehicle on the scan matching process. In phase II, the method divides the entire map into several 2D maps based on the height of the MAV. Each 2D map has its own height and it covers part of the environment, as well as a transient region between each two maps to enable connecting neighboring maps as shown in Figure 1. This splitting process aims at reducing the memory consumption and processing costs of the entire constructed map and avoiding restrictions for assigning grid size before the flight. Moreover, for more memory reduction, the proposed method retains the executed trajectory for backward return, instead of building another 2D map slices during the exit process.

This paper is organized as follows: the used methodology is explained in two phases. Section 2 demonstrates the first phase for roll and pitch compensation, while the

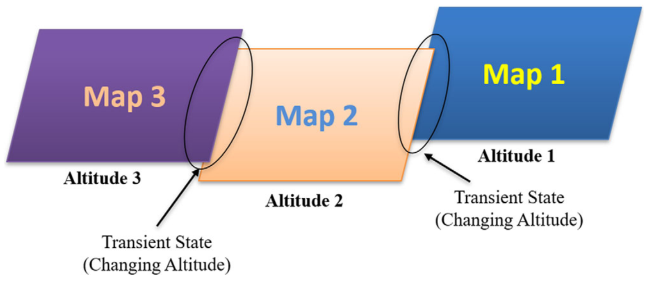
Map 3
Map 2
Map 1
Altitude 3
Altitude 2
Altitude 1
Transient State (Changing Altitude)
Transient State (Changing Altitude)

FIGURE 1 Transient regions between several 2D map slices. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

second phase, which is the multi-slices representation, is illustrated in Section 3. Section 4 illustrates the hardware system architecture of the employed MAV. The experimental results are presented in Section 5. Finally, the conclusion is provided in Section 6.

2 | PHASE I: ROLL AND PITCH COMPENSATION

Figure 2 demonstrates the overall structure of the roll and pitch compensation approach. The roll and pitch angles of the MAV are calculated from the accelerometer measurements (f_x, f_y). These angles are used to calculate the rotation matrix (R_{level}) and consequently level the current scan in the body frame (b-frame). The current leveled scan in the b-frame together with the previous leveled scan in the mapping frame (m-frame) are fed to the improved reference key frame (IRKF) algorithm (Mohamed, Moussa, Elhabiby, & El-Sheimy, 2019) to perform the scan matching process. The IRKF algorithm estimates position and heading (P_x, P_y, θ) of the MAV in the m-frame.

Equation (1) presents the model equation of the measurements of the specific force (Noureldin et al., 2013):

$$I_f = f_t + b_f + S_f f_t \tag{1}$$

$$S_f = \begin{bmatrix} s_{ix} & 0 & 0 \\ 0 & s_{iy} & 0 \\ 0 & 0 & s_{iz} \end{bmatrix}, \tag{2}$$

where

I_f: the vector of the accelerometer measurement (m/s^2)
f_t: the vector of the true specific force vector (m/s^2)
b_f: the bias of the accelerometer instrument (m/s^2)
S_f: the matrix of the linear scale factor error (ppm)
s_{ix}: the linear scale factor error in the x-direction (ppm)
s_{iy}: the linear scale factor error in the y-direction (ppm)
s_{iz}: the linear scale factor error in the z-direction (ppm).

The deterministic errors (i.e., bias, scale factor) are calibrated using the six-position static test. Therefore, the specific force vector after the correction of the deterministic errors is expressed by

$$f_{corrected} = (I_f - b_f)/\left(1 + S_f\right), \tag{3}$$

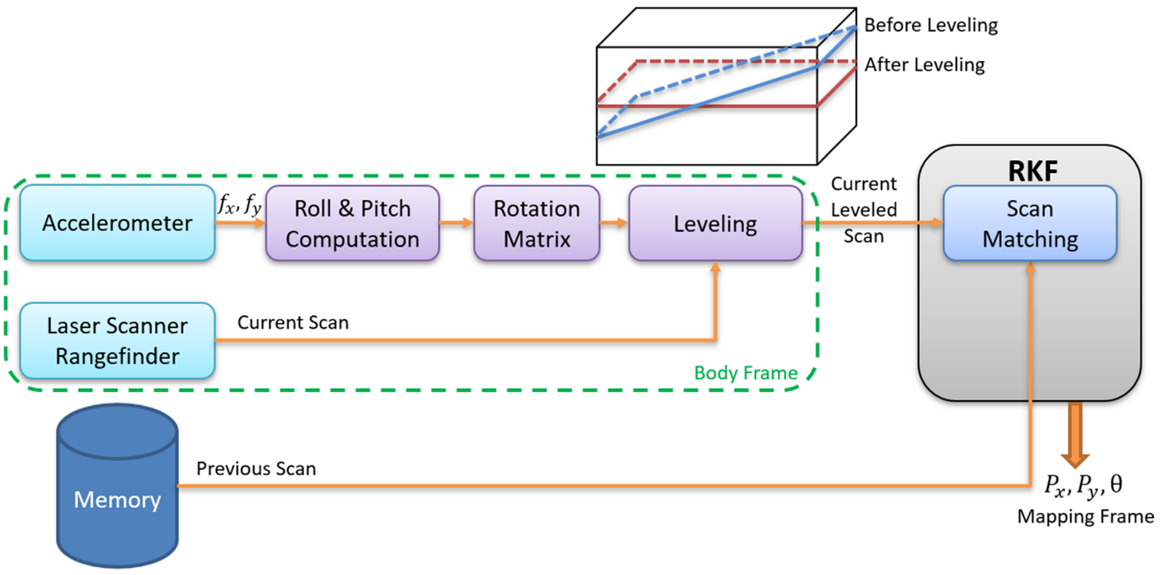

FIGURE 2 Overall structure of the roll and pitch compensation approach. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

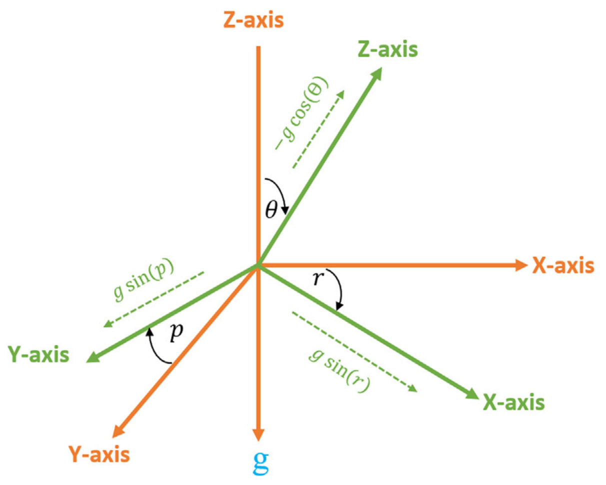

FIGURE 3 The computation of the pitch (p) and roll (r) angles using accelerometer measurement. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

where $f_{corrected}$: the corrected specific force from the deterministic errors.

Figure 3 shows the computation of the pitch (p) and roll (r) angles using the accelerometer measurement. Due to the gravitational field and the tilting status of the accelerometer, the accelerometer measures components of the gravity (g) as follows (Noureldin et al., 2013):

$$\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = \begin{bmatrix} -g\cos(p)\sin(r) \\ g\sin(p) \\ g\cos(p)\cos(r) \end{bmatrix}, \tag{4}$$

where

f_x: the accelerometer measurement in the x-direction
f_y: the accelerometer measurement in the y-direction
f_z: the accelerometer measurement in the z-direction
r: the tilting angle (roll) around the y-direction
p: the tilting angle (pitch) around the x-direction
g: the magnitude of the gravitational field.

From Equations (3) and (4):

$$roll = r = \tan^{-1}\left(\frac{-f_{x-corrected}}{f_{z-corrected}}\right) \tag{5}$$

$$pitch = p = \tan^{-1}\left(\frac{f_{y-corrected}}{\sqrt{f_{x-corrected}{}^2 + f_{z-corrected}{}^2}}\right), \tag{6}$$

where $f_{x-corrected}$: the corrected specific force from the deterministic errors in the x-direction

$f_{y-corrected}$: the corrected specific force from the deterministic errors in the y-direction
$f_{z-corrected}$: the corrected specific force from the deterministic errors in the z-direction.

Therefore, the output of the accelerometer in the x and z directions can be used to compute the tilt angle (roll) around the y-axis, as given in Equation (5). Similarly, the output of the accelerometer in the three (i.e., x, y, and z) directions can be used to compute the tilt angle (pitch) around the x-axis, as given in Equation (6).

After the roll and pitch are computed and the azimuth angle is set to zero, the true azimuth will be estimated from the next scan matching step. The rotation matrix (R_{level}) is computed as follows:

$$R_{level} = \begin{bmatrix} \cos(r) & 0 & \sin(r) \\ \sin(p)\sin(r) & \cos(p) & -\sin(p)\cos(r) \\ -\cos(p)\sin(r) & \sin(p) & \cos(p)\cos(r) \end{bmatrix}, \tag{7}$$

where

ρ: pitch angle
r: roll angle.

The current leveled scan is computed using the rotation matrix (R_{level}) as follows:

$$(P_{cl})_{3xn} = (R_{level})_{3x3}\,(P_{cs})_{3xn} \tag{8}$$

$$(P_{cs})_{nx3} = \begin{bmatrix} P_{cxi} & P_{cyi} & 1 \\ \vdots & \vdots & \vdots \end{bmatrix}, \tag{9}$$

where

P_{cl}: the current leveled scan
P_{cs}: the current scan
P_{cxi}: x-coordinate of current scan i^{th} point
P_{cyi}: y-coordinate of current scan i^{th} point.

3 | PHASE II: MULTI-SLICES OF 2D POINT CLOUD FOR 3D ENVIRONMENT REPRESENTATION

The autonomous system of the MAV consists of two subsystems, namely: navigation and autopilot subsystems.

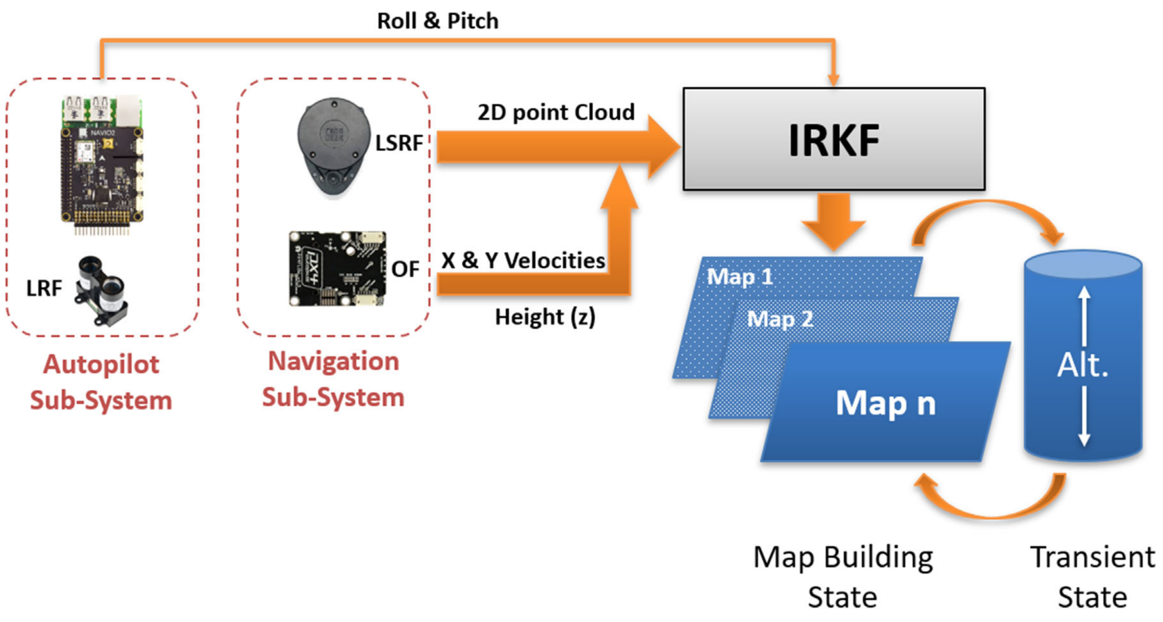

FIGURE 4 Overall structure of the proposed multi-slices 2D point cloud. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

The autopilot subsystem includes flight controller and single beam laser rangefinder (LRF), while the navigation subsystem includes laser scan rangefinder (LSRF) and optical flow (OF). Figure 4 illustrates the overall structure of the proposed multi-slices 2D point cloud for 3D environment representation. The utilized IRKF algorithm is fed with the 2D point clouds (i.e., previous and current leveled scans), the height of the MAV from the ground, and the velocities of the MAV in the x and y directions with respect to the b-frame from the navigation subsystem, while the roll and pitch of the MAV are provided from the autopilot subsystem.

The proposed multi-slices method consists of multi-level 2D point clouds separated by vertical distance. The method creates a 2D point cloud for each period of time without a significant height difference ($\pm 10\ cm$); when the MAV is forced to change the height beyond the threshold, a new slice of 2D point cloud is created, and a transient state is initiated to serve the hand-over between the previous and the new slices. This process iterates throughout the entire flight.

The proposed multi-slices method comprises two states, namely: map-building and transient states, and two transmissions, namely: forward and backward transmissions as shown in Figure 5. The transmission between the states is carried out after reaching switching criteria related to the detected gaps in the current scene. The candidate gaps are chosen according to two factors. Primarily, the laser scanner rangefinder measurements fail to get returns from the around obstacles. The dimension of the detected gap is the second factor of validity, wherein the distance, from

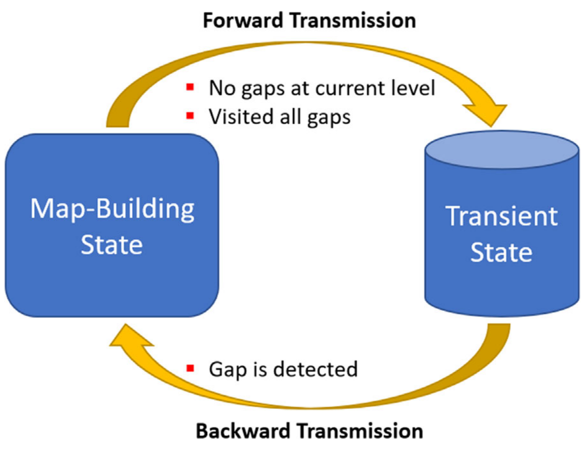

FIGURE 5 The two states of the proposed method with the transmissions between them. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

the vehicle to each side around it, is not less than ($25\ cm$). The MAV starts building map at an initial height using the IRKF algorithm. Hence, the IRKF algorithm performs the scan matching process between the current leveled scan from phase I and the previous scan. When no gaps are detected at the current height of the MAV or when all the existing gaps are already visited by the MAV, the current map-building state is quitted. Then the proposed method is switched to the transient state.

The transient state handles the task of determining the relation between old and new maps. During the transient state, the map construction is halted, and the new map

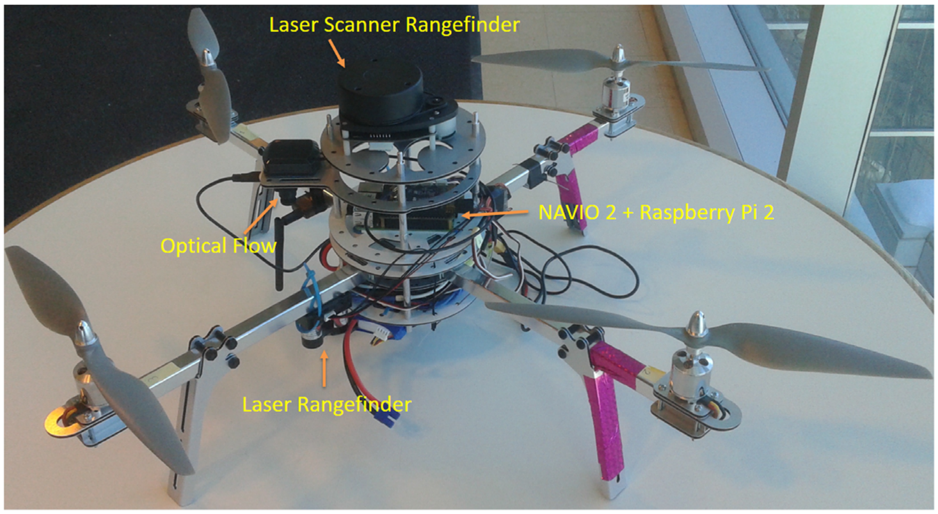

FIGURE 6 Hardware system architecture. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

is built from scratch without a history guide. The MAV seeks new gaps by changing its height up and down to detect a new gap. The scan matching process is implemented between each of the two successive scan frames until the end of the transient state in order to estimate the transformation parameters between the previous constructed map and the next map using the utilized IRKF algorithm. If a gap is detected at a new height, the proposed method is switched back to the map-building state to construct a new map at the new height. Thus, the entire map is represented as several 2D point cloud slices at different heights. If there are no detected gaps during the transient state, the MAV stops building more maps for the environment and uses the implemented trajectory for backward return.

4 | MAV'S HARDWARE SYSTEM ARCHITECTURE

Figure 6 illustrates the proposed hardware system architecture. In addition to the aerial platform (i.e., x-frame quadcopter), the system architecture consists of two subsystems, namely: navigation and autopilot subsystems. The main components of the navigation subsystem are a low-cost 2D laser scanner rangefinder (LSRF) and an optical flow (OF) sensor attached with a sonar. On the other hand, the main components of the autopilot subsystem are the single beam laser rangefinder (LRF) and flight controller, consisting of the Raspberry Pi 2 model B embedded computer and NAVIO 2 autopilot. The used LSRF is RPLIDAR 360° from RoboPeak. This sensor provides 2D point cloud data that is characterized by short range detection, and Table 1 presents the sensor specifications.

TABLE 1 The specifications of the utilized laser scanner rangefinder

Maximum detection range	6 [m]
Minimum detection range	0.2 [m]
Max scan rate (rotation speed)	7 Hz
Field of view	360°
Angular resolution at maximum rotation speed	1.5°

The utilized OF sensor is PX4FLOW (pixhawk) (Honegger et al., 2013). This sensor consists of imaging sensor (camera), ultrasonic rangefinder (sonar), and low-cost MEMS three-axes gyro. The employed LRF sensor is PulsedLight http://static.garmin.com/pumac/LIDAR_Lite_v3_Operation_Manual_and_Technical_Specifications.pdf. The maximum detection range of this sensor is 40 m with the accuracy of +/- 0.025 m and frequency of 500 Hz.

The LSRF is mounted horizontally on the top of the MAV; the LRF is installed orthogonally to the LSRF and fixed at the bottom pointing to the ground. The OF sensor is mounted at the bottom pointing to the ground as well. The autopilot (i.e., NAVIO 2) is equipped with the dual IMU (MPU9250 9DOF and LSM9DS1 9DOF) https://cdn.sparkfun.com/assets/learn_tutorials/5/5/0/MPU9250REV1.0.pdf, https://cdn.sparkfun.com/assets/learn_tutorials/3/7/3/LSM9DS1_Datasheet.pdf, magnetometer, barometer (MS5611) https://www.te.com/commerce/DocumentDelivery/DDEController?Action=srchrtrv&DocNm=MS5611-01BA03&DocType=Data+Sheet&DocLang=English, and U-blox M8N GPS/Glonass/Beidou receiver.

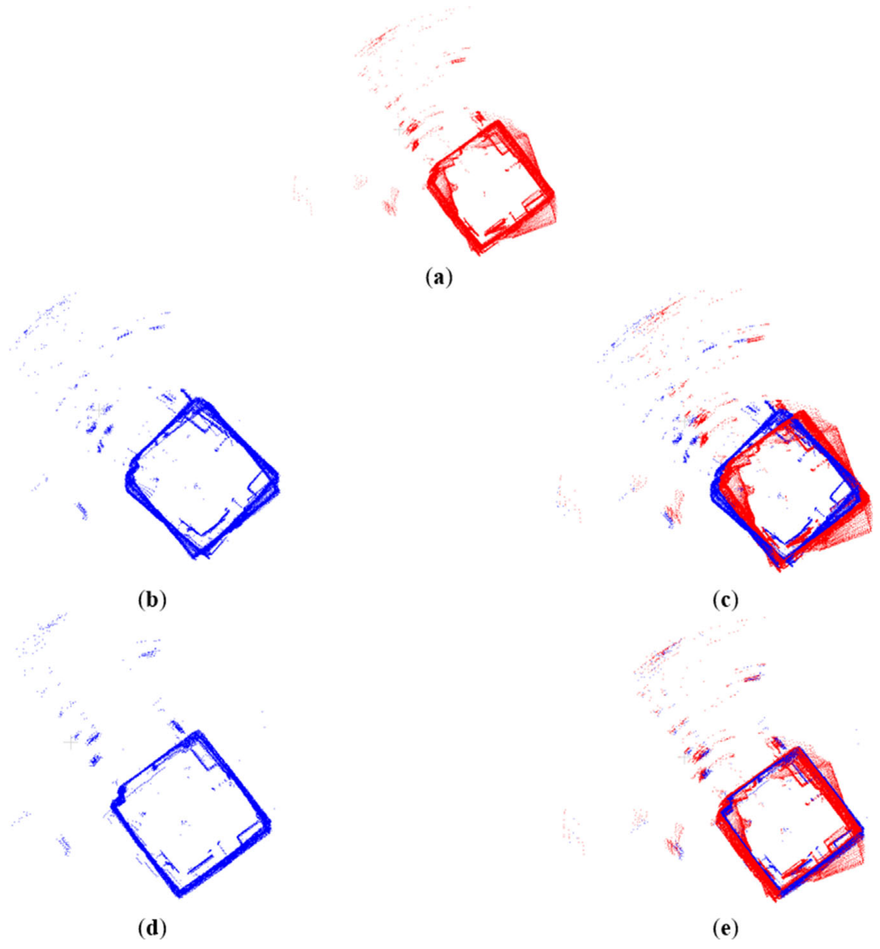

FIGURE 7 The scan matching result of the transient state using different algorithms. a) Raw point cloud b) Corners registration method c) Merging the raw point cloud with the result of the corner registration method d) IRKF algorithm e) Merging the raw point cloud with the result of the IRKF algorithm. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

5 | EXPERIMENTAL RESULTS

As mentioned earlier, during the transient state, the scan matching process is implemented to estimate the transformation parameters between each two neighboring maps. Therefore, the scan matching performance of the transient state is evaluated by estimating the scan matching error. This is performed by computing the RMSE of the matched points for each two successive scans over the transient period (Mohamed, Moussa, Elhabiby, El-Sheimy, & Sesay, 2016). Although feature-to-feature scan matching methods are more robust than point-to-point scan matching methods (Yin et al., 2014), the opportunity of finding the same feature from scan to scan decreases due to the vertical translation of the vehicle. Thus, for performance assessment, the transient state is tested with various point-to-point and feature-to-feature methods such as the iterative closest point (ICP), corners registration (Mohamed et al., 2016), and IRKF algorithm.

Figure 7 shows the scan matching result of the transient state using different algorithms. Subplot (a) shows the raw point cloud without processing. Subplot (b) presents the result of the scan matching using corners registration, while both subplots (i.e., a & b) are merged in subplot (c).

TABLE 2 The average RMSE between each two successive scans using different algorithms

Method / Algorithm	ICP Algorithm	Corners Registration Method	IRKF Algorithm
Average RMSE [cm]	4.0	4.1	1.8

Subplot (d) illustrates the result of the scan matching using the IRKF algorithm, while both subplots (i.e., a & d) are merged in subplot (e).

The average RMSEs of the tested transient state are 4.0, 4.1, and 1.8 cm using the ICP algorithm, corners registration, and IRKF algorithm, respectively, as demonstrated in Table 2. It is clear that the IRKF algorithm obtains the average RMSE less than the ICP algorithm and corners registration by approximately 55% and 56%, respectively. Therefore, the IRKF algorithm is chosen to estimate the transformation parameters during the transient state.

Figure 8 illustrates 3D mapping representation for the same environment using various methods and approaches. Subplot (a) shows the solution of the standard 3D point cloud, while the solution of the voxel occupancy grids is presented in subplot (b). Subplot (c) presents the

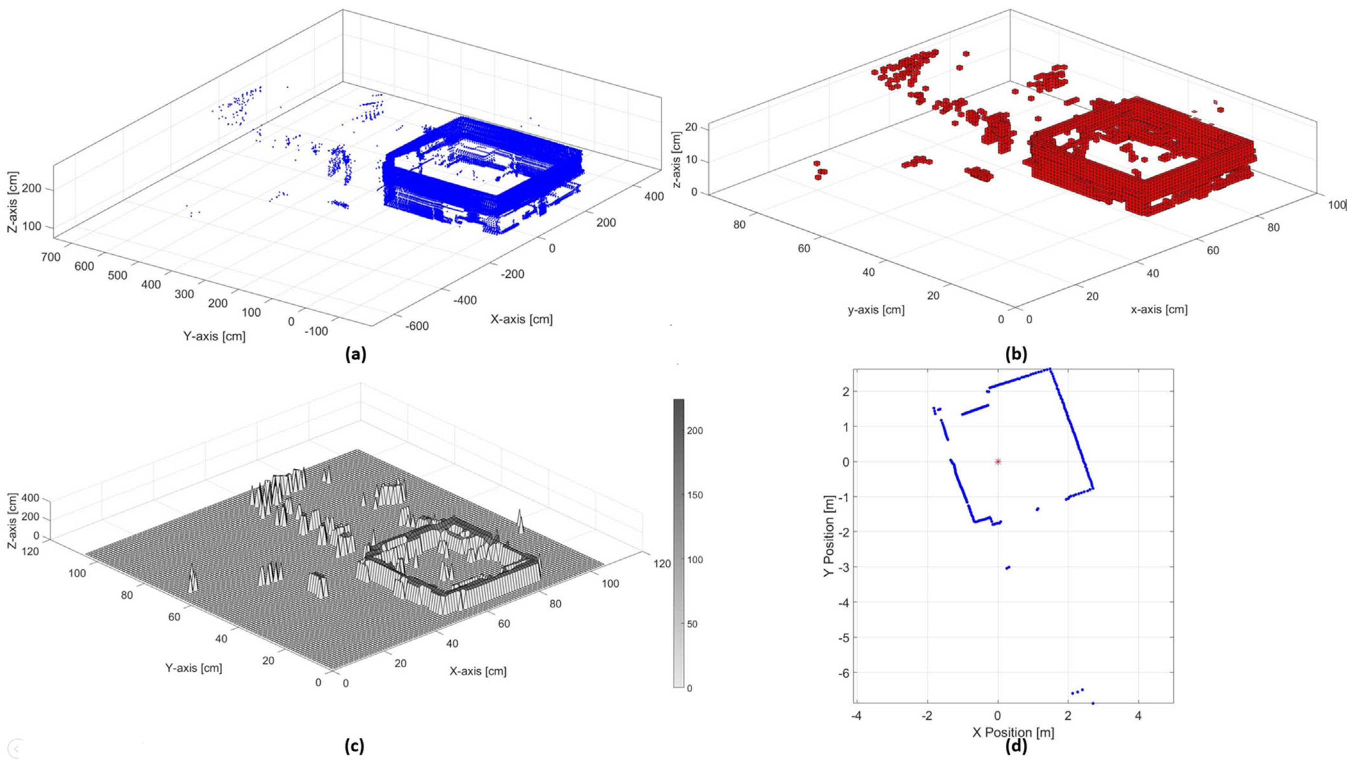

FIGURE 8 3D mapping representation for the same environment using various methods and approaches. a) Standard point cloud b) Voxel occupancy grids c) Elevation map approach d) Proposed method. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

TABLE 3 A comparison of memory consumption between different methods

Method/Algorithm	Standard point cloud	Voxel	Elevation map	Proposed method
Size of the environment	$10.6 \times 10.6 \times 3.0\ [m^3]$ (width x length x height)			
Size of the cell and Voxel in the Occupancy Grid	–	$10 \times 10 \times 10\ [cm^3]$	$10 \times 10\ [cm^2]$	–
Memory consumption [bytes]	773.5 K	2.6 M	86.8 K	**4.7 K**
Percentage of reduction w.r.t. the proposed method	99.39%	99.83%	94.70%	–

solution of the elevation map approach, the height of each cell comes from averaging all heights of all points that fall in the same cell, while subplot (d) shows the solution of the proposed method. The scanning process is performed in an office room.

Table 3 demonstrates a comparison of memory consumption between standard point cloud, voxel occupancy grids, the elevation map approach, and the proposed method. The width, length, and height of the tested environment are 10.6m, 10.6m, and 3.0m, respectively. The width, length, and height of the utilized voxel are 10cm, 10cm, and 10cm, respectively, while the cell size of the occupancy grid is 10cm x 10cm. The memory consumption of the standard point cloud, voxel occupancy grids, elevation map approach, and the proposed method are 773.5KB, 2.6MB, 86.8KB, and 4.7KB, respectively. Thus, the proposed method reduces the memory consumption by approximately 164x, 553x, and 18x, respectively, w.r.t. the standard point cloud, voxel occupancy grids, and finally the elevation map approach.

Figure 9 shows a plan view of the first dataset: ENE building at University of Calgary, using the multi-slices 2D point cloud approach. Two maps are constructed for the entire mission, and each map has a fixed height. The green points represent the first map (Map-1), while the second map (Map-2) is represented by the blue points. The red asterisks represent the trajectory of the MAV. The black arrow and the dashed rectangle show the location of the transient state.

Figure 10 illustrates the 3D representation of the first dataset using the multi-slices 2D point cloud approach. The dashed rectangle shows that the MAV is getting stuck

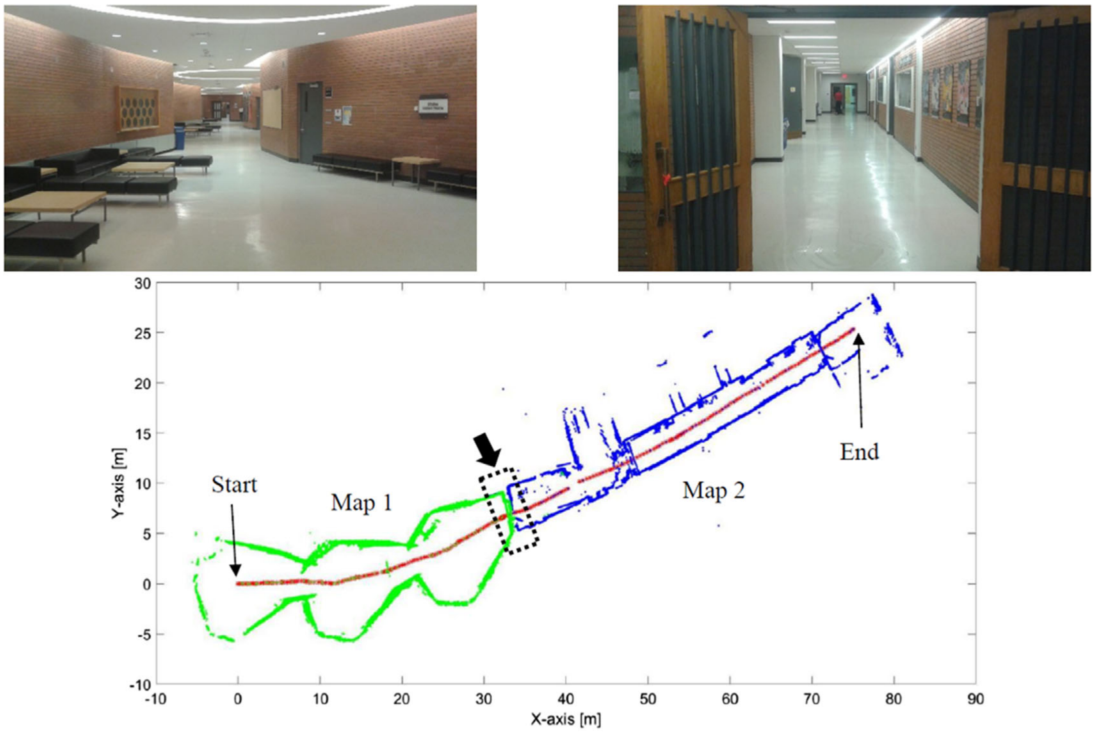

FIGURE 9 Plan view of the first dataset using multi-slices 2D point cloud. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

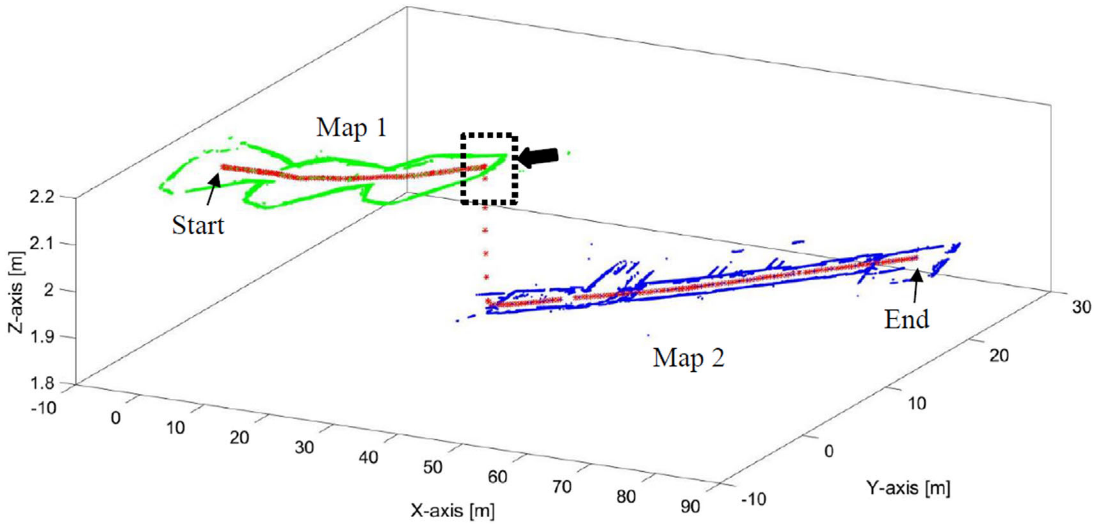

FIGURE 10 3D representation of the first dataset using multi-slices 2D point cloud. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

because there are no gaps except a previously visited gap. Thus, the proposed method terminates the map-building state (Map-1) at a height of 2.2 m and switches to the transient state. The MAV decreases the height in order to find a new gap. Once a new gap is detected, the method switches back to the map-building state to construct the next map (Map-2) with the current height, 1.8 m.

Figure 11 illustrates the height of the MAV for the entire mission of the first dataset. After performing the process of the take-off and reaching the predefined height according to the environment, the MAV starts to construct the first map (Map-1) while maintaining the approximately fixed height. Once the MAV is trapped due to no gaps being detected in the current height, the MAV switches to the

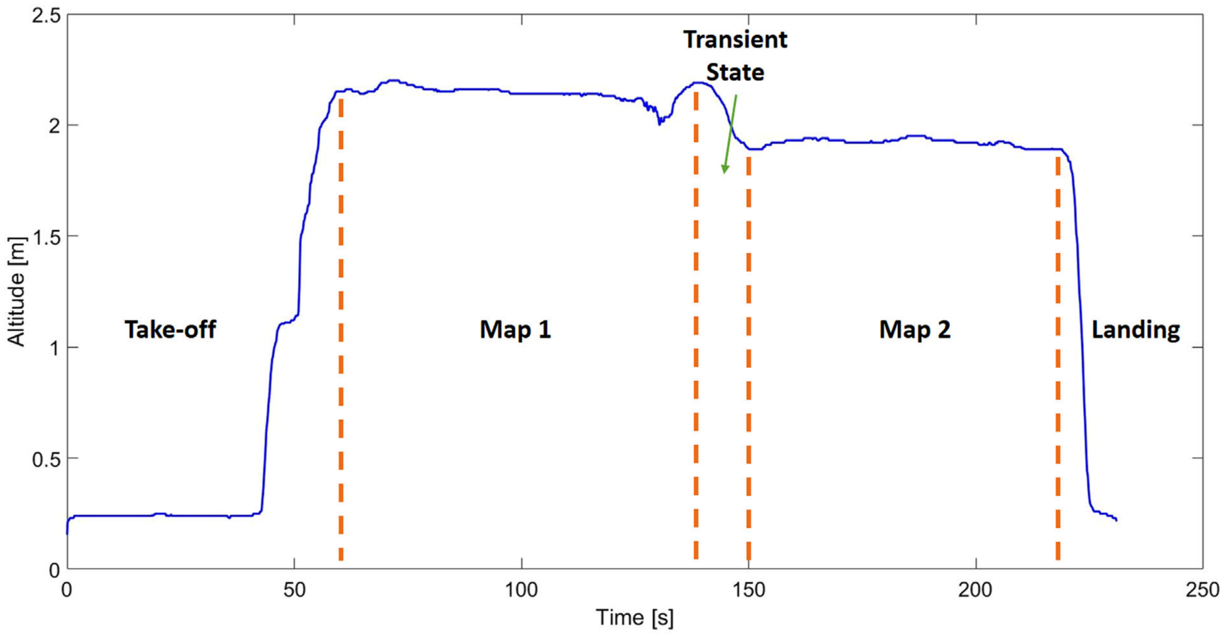

FIGURE 11 The height of the MAV during the first test. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

transient state and decreases its height hoping to detect new gaps. The MAV returns to the map-building state to construct the second map (Map-2). After finishing the mission, the MAV reduces its height to land.

Figure 12 shows a plan view of the second dataset: CCIT building at University of Calgary, using the multi-slices 2D point cloud approach. Two maps are also constructed for the entire mission and each map has a fixed height. The green points represent the first map (Map-1), while the second map (Map-2) is represented by the blue points. The red asterisks represent the trajectory of the MAV. The black arrow and the dashed rectangle show the location of the transient state.

Figure 13 illustrates the 3D representation of the second dataset using the multi-slices 2D point cloud approach. The dashed rectangle shows that the MAV is getting stuck because there are no gaps. Thus, the proposed method terminates the map-building state (Map-1) at the height of 2.2 m and switches to the transient state. The MAV decreases the height in order to find a new gap. Once a new gap is detected, the method switches back to the map-building state to construct the next map (Map-2) with the current height, 1.8 m.

Figure 14 illustrates the height of the MAV for the entire mission of the second dataset. The MAV increases the thrust of its motors to perform the take-off procedure. After performing the take-off process and reaching the required height according to the environment, the MAV starts to construct the first map (Map 1) while preserving approximately the same height. Due to no gaps being detected in the current height, the MAV switches to the transient state and decreases its height trying to detect new gaps. The MAV returns to the map-building state to construct the second map (Map 2). After finishing the mission, the MAV reduces its height to land.

6 | CONCLUSION

The autonomous navigation of unmanned vehicles in an unknown indoor environment is typically addressed by the simultaneous localization and mapping (SLAM) approach. In contrast to 2D-mapping representation, the 3D mapping requires high-processing computational and memory costs. These high costs can limit using the low-cost embedded systems in the unmanned vehicles. Consequently, different approaches were proposed to decrease the computational and memory costs such as the elevation map, Octomap, and multi-level occupancy grids. However, these approaches still suffer from computational and memory costs. For exploration tasks, that are performed in search and rescue operations, the mapping process is a means to achieve the navigation solution in such environments. Furthermore, the vehicle is vulnerable to loss in search and rescue operations due to hazardous situations. Therefore, a real-time 3D SLAM approach, based on the low-cost 2D laser scanner rangefinder, is proposed for indoor navigation. The proposed approach is divided into two phases. The role of phase one is to level the current point cloud to mitigate the effect of the MAVs orientation (i.e., roll and

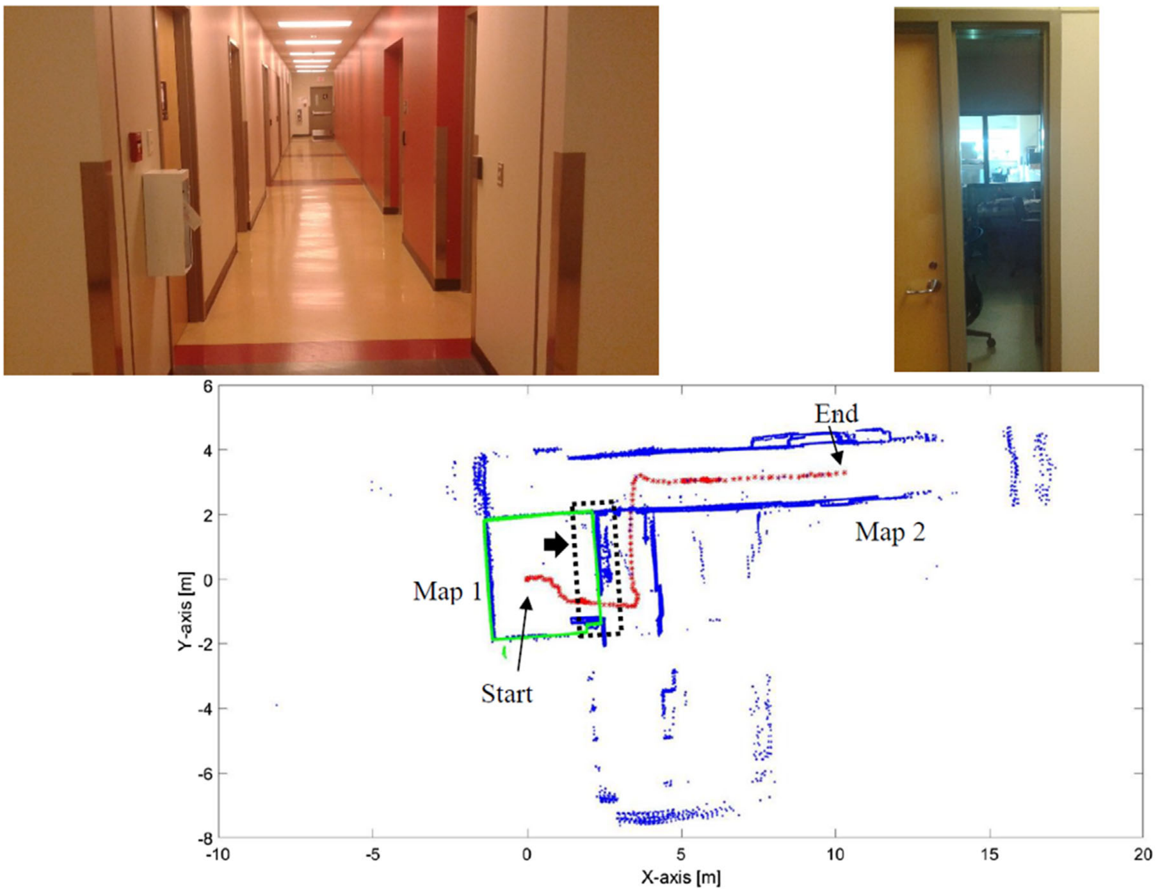

FIGURE 12 Plan view of the second dataset using multi-slices 2D point cloud. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

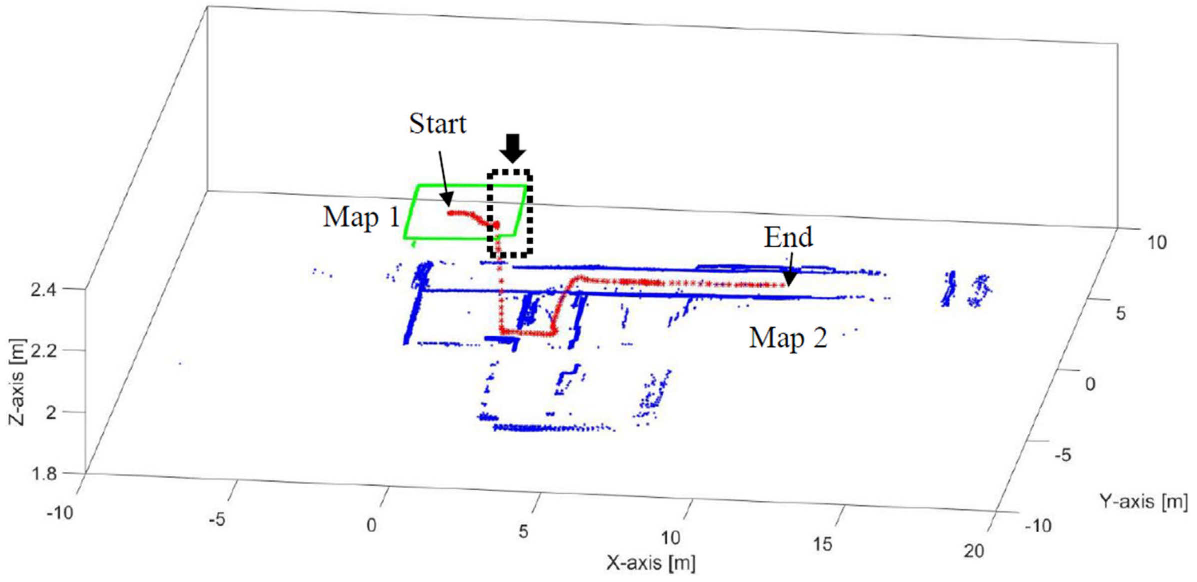

FIGURE 13 3D representation of the second dataset using multi-slices 2D point cloud. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

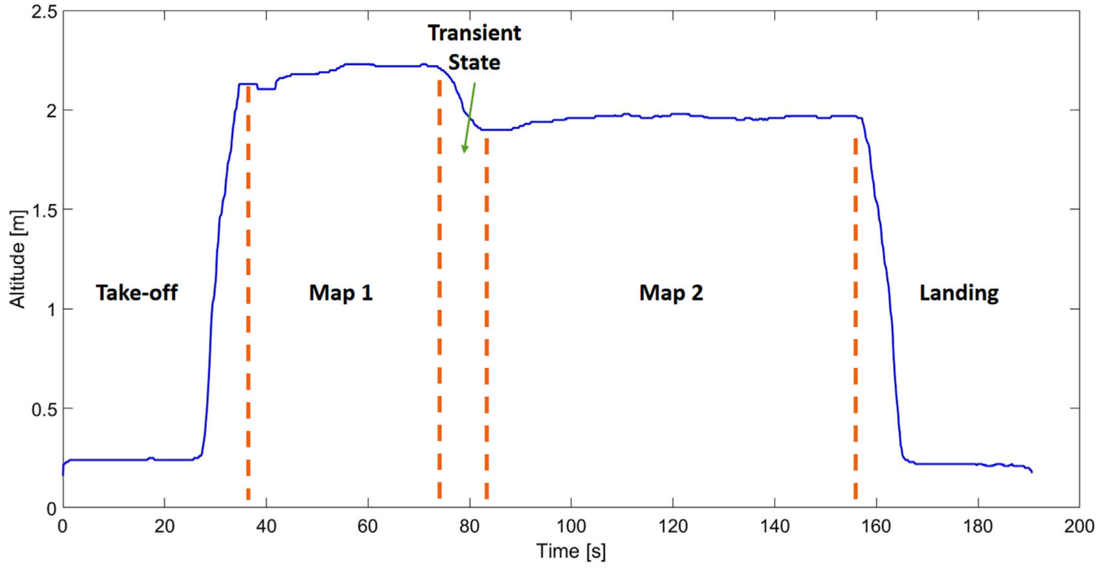

FIGURE 14 The height of the MAV during the second test. [Color figure can be viewed in the online issue, which is available at wiley-onlinelibrary.com and www.ion.org]

pitch angles). In phase two, instead of collecting huge data, the entire environment is divided into several 2D maps based on the height of the MAV. Thus, each 2D map has its own height and it covers part of the environment, as well as a transient region between each two successive maps to enable connecting these maps together. This splitting process aims to reduce the memory consumption and the processing costs of the entire constructed map and to avoid restrictions for assigning grid size before the flight. The map construction is performed using the IRKF algorithm. The estimation of the transformation parameters during the transient region is implemented using the IRKF as well.

For the transient state, the scan matching results, using different methods and algorithms, are tested in order to choose the appropriate method as one of the transient state roles is to accurately connect the neighboring maps together. The average RMSE between each two successive scans is 4.0, 4.1, and 1.8 cm using the ICP algorithm, corners registration, and IRKF algorithm, respectively. The performance of the proposed method is tested by computing the memory consumption of the same environment using various methods and approaches such as the standard 3D point cloud, voxel occupancy grids, and elevation map approach. The proposed method succeeds to reduce the memory consumption by 99.39%, 99.83%, and 94.70% among standard 3D point cloud, voxel occupancy grids, and elevation map approaches, respectively.

ACKNOWLEDGMENTS

This work was supported by Dr. Naser El-Sheimy research funds from NSERC and Canada Research Chairs programs. Thanks also goes to the funding of the first author by the Egyptian government.

ORCID

H. A. Mohamed https://orcid.org/0000-0003-2237-7995

REFERENCES

Alpen, M., Frick, K., & Horn, J. (2012). A real-time on-board orthogonal SLAM for an indoor UAV. *Intelligent Robotics and Applications*, *7508*, 542–551. https://doi.org/10.1007/978-3-642-33503-7_53

Anderson-Sprecher, P., & Simmons, R. (2012, May). Voxel-based motion bounding and workspace estimation for robotic manipulators. *2012 IEEE International Conference on Robotics and Automation*, Saint Paul, MN, 2141–2146. https://doi.org/10.1109/ICRA.2012.6225256

Dryanovski, I., Morris, W., & Xiao, J. (2010, October). Multi-volume occupancy grids: An efficient probabilistic 3D mapping model for micro aerial vehicles. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, Taiwan, 1553–1559. https://doi.org/10.1109/IROS.2010.5652494

Duffy, N., Allan, D., & Herd, J. T. (1989). Real-time collision avoidance system for multiple robots operating in shared work-space. *IEE Proceeding E – Computers and Digital Techniques*, *136*(6), 478–484. https://doi.org/10.1049/ip-cdt:20040043

Gageik, N., Benz, P., & Montenegro, S. (2015). Obstacle detection and collision avoidance for a UAV with complementary low-cost sensors. *IEEE Access*, *3*, 599–609. https://doi.org/10.1109/ACCESS.2015.2432455

Hadsell, R., Bagnell, J. A., & Hebert, M. (2009, June). Accurate rough terrain estimation with space-carving kernels. *Proceedings of the Robotics: Science and Systems (RSS)*, Seattle, WA. https://doi.org/10.15607/RSS.2009.V.019

Herbert, M., Caillas, C., Krotkov, E., Kweon, I. S., & Kanade, T. (1989, May). Terrain mapping for a roving planetary explorer. *Proceedings of the IEEE International Conference on Robotics and Automation*

(ICRA), Scottsdale, AZ, 997–1002. https://doi.org/10.1109/ROBOT.1989.100111

Honegger, D., Meier, L., Tanskanen, P., & Pollefeys, M. (2013, May). An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications. *Proceedings of the IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, 1736–1741. https://doi.org/10.1109/ICRA.2013.6630805

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., & Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, *34*, 189–206. https://doi.org/10.1007/s10514-012-9321-0

Jessup, J., Givigi, S. N., & Beaulieu, A. (2014, October). Robust and efficient multi-robot 3D mapping with octree based occupancy grids. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, San Diego, CA, 3996–4001. https://doi.org/10.1109/SMC.2014.6974556

Kalogeiton, V. S., Ioannidis, K., Sirakoulis, G. C., & Kosmatopoulos, E. B. (2019). Real-time active SLAM and obstacle avoidance for an autonomous robot based on stereo vision. *Cybernetics and System*, *50*, 239–260. https://doi.org/10.1080/01969722.2018.1541599

Kamarudin, K., Mamduh, S.M., Yeon, A.S.A., Visvanathan, R., Shakaff, A.Y.M., Zakaria, A., Kamarudin, L.M., & Rahim, N.A. (2015, October). Improving performance of 2D SLAM methods by complementing Kinect with laser scanner. *Proceedings of the IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*, Langkawi, Malaysia, 278–283. https://doi.org/10.1109/iris.2015.7451625

Khan, S., Wollherr, D., & Buss, M. (2015, May). Adaptive rectangular cuboids for 3D mapping. *Proceedings of the IEEE International Conference on Robotics and Automation*, Seattle, WA, 2132–2139. https://doi.org/10.1109/ICRA.2015.7139480

Kohlbrecher, S., von Stryk, O., Meyer, J., & Klingauf, U. (2011, November). A flexible and scalable SLAM system with full 3D motion estimation. *Proceedings of the IEEE International Symposium on Safety, Security, and Rescue Robotics*, Kyoto, Japan, 155–160. https://doi.org/10.1109/SSRR.2011.6106777

Mohamed, H. A., Moussa, A. M., Elhabiby, M. M., El-Sheimy, N., & Sesay, A. B. (2016, June). Improved real-time scan matching using corner features. *Proceedings of the International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Prague, Czech Republic, 533–539. https://doi.org/10.5194/isprsarchives-XLI-B5-533-2016

Mohamed, H. A., Moussa, A., Elhabiby, M. M., & El-Sheimy, N. (2019, June). Improved reference key frame algorithm. *Proceedings of the ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Enschede, Netherland, 133–139. https://doi.org/10.5194/isprs-annals-IV-2-W5-133-2019

Morris, W., Dryanovski, I., & Xiao, J. (2010, June). 3D indoor mapping for micro-UAVs using hybrid range finders and multi-volume occupancy grids. *Proceedings of the RSS 2010 Workshop RGB-D Advanced Reasoning with Depth Cameras*, Zaragoza, Spain, 743–747. https://doi.org/10.1016/j.mehy.2007.08.013

Noureldin, A., Karamat, T. B., & Georgy, J. (2013). Inertial navigation system. In *Fundamentals of inertial navigation, satellite-based positioning and their Integration* (pp. 125–166). https://doi.org/10.1007/978-3-642-30466-8_4

Pfaff, P., & Burgard, W. (2006). An efficient extension of elevation maps for outdoor terrain mapping. In P. Corke & S. Sukkariah (Eds.), *Field and service robotics: Results of the 5th international conference* (pp. 195–206). https://doi.org/10.1007/978-3-540-33453-8_17

Pfaff, P., Triebel, R., & Burgard, W. (2007). An efficient extension to elevation maps for outdoor terrain mapping and loop closing. *The International Journal of Robotics Research*, *26*(2), 217–230. https://doi.org/10.1177/0278364906075165

Plaza-Leiva, V., Gomez-Ruiz, J. A., Ababsa, F.-E., Mandow, A., Morales, J., & Garcia-Cerezo, A. (2015, June). Occupancy grids generation based on geometric-featured voxel maps. *Proceedings of 23rd Mediterranean Conference on Control and Automation (MED)*, Torremolinos, Spain, 766–771. https://doi.org/10.1109/MED.2015.7158838

Rivadeneyra, C., Miller, I., Schoenberg, J. R., & Campbell, M. (2009, May). Probabilistic estimation of multi-level terrain maps. *Proceedings of the IEEE International Conference on Robotics and Automation*, Kobe, Japan, 1643–1648. https://doi.org/10.1109/ROBOT.2009.5152767

Souza, A., & Gonçalves, L. M. (2016). Occupancy-elevation grid: An alternative approach for robotic mapping and navigation. *Robotica*, *34* (11), 2592–2609. https://doi.org/10.1017/S0263574715000235

Thrun, S. (2007). Simultaneous localization and mapping. In M. E. Jefferies & W.-K. Yeap (Eds.), *Robotics and cognitive approaches to spatial mapping* (Vol. *38*, pp. 13–41). https://doi.org/10.1007/978-3-540-75388-9_3

Tian, Y., Huang, W., Wang, Y., Yi, X., Wang, Z., & Yang, X. (2016). Multi-level occupancy grids for efficient representation of 3D indoor environments. In R. Booth & M.L. Zhang (Eds.), *PRICAI 2016: Trends in artificial intelligence* (Vol. *9810*, pp. 517–528). https://doi.org/10.1007/978-3-319-42911-3_43

Triebel, R., Pfaff, P., & Burgard, W. (2006, October). Multi-level surface maps for outdoor terrain mapping and loop closing. *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, Beijing, China, 2276–2282. https://doi.org/10.1109/IROS.2006.282632

Wurm, K. M., Hornung, A., Bennewitz, M., Stachniss, C., & Burgard, W. (2010, January). OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. *Proceedings of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation*, *2*, 403–412. http://ais.informatik.uni-freiburg.de/publications/papers/wurm10octomap.pdf

Yin, J., Carlone, L., Rosa, S., & Bona, B. (2014, August). Graph-based robust localization and mapping for autonomous mobile robotic navigation. *Proceedings of the IEEE International Conference on Mechatronics and Automation*, Tianjin, China, 1680–1685. https://doi.org/10.1109/ICMA.2014.6885953

Zhang, J., Liu, S., Gao, B., & Zhong, C. (2018, June). An improvement algorithm for OctoMap based on RGB-D SLAM. *2018 Chinese Control and Decision Conference (CCDC)*, Shenyang, China, 5006–5011. https://doi.org/10.1109/CCDC.2018.8407999

How to cite this article: Mohamed HA, Moussa AM, El-Sheimy N, Elhabiby MM. Multi-slices navigation approach for unknown 3D environments using micro aerial vehicles. *NAVIGATION*. 2021;68:61–73. https://doi.org/10.1002/navi.403