An Autonomous Mapping Approach for Confined Spaces using Flying Robots

Ahmad Alsayed[1,2], Mostafa R. A. Nabawy[1,3], Akilu Yunusa-Kaltungo[1], Mark K. Quinn[1], and Farshad Arvin[4]

[1] Department of Mechanical, Aerospace and Civil Engineering, The University of Manchester, Manchester M1 3BB, UK
[2] Department of Mechanical Engineering, Umm Al-Qura University, Al Abdeyah, Makkah 5555, Saudi Arabia
[3] Aerospace Engineering Department, Faculty of Engineering, Cairo University, Giza 12613, Egypt
[4] Department of Electrical and Electronic Engineering, The University of Manchester, Manchester M1 3BB, UK
ahmad.alsayed@manchester.ac.uk

Abstract. Mapping a confined space with a drone-based system becomes challenging when vision sensors cannot be used due to environmental constraints. This paper presents a novel scan-matching approach based on an Iterative Closest Point algorithm that uses low-rate and low-dense scans from a LiDAR. The proposed technique only employs the horizontal layer from a 3D LiDAR to estimate the transformation matrices in a computationally efficient fashion, which is then used to generate the 3D map of the scanned environment in real-time. This is, then, complemented with a fit-for-purpose indoor navigation path-planning strategy. The method was successfully tested by mapping a confined space within a cement plant simulated environment and estimating a stockpile volume stored in that space. The volume of the reconstructed stockpile was estimated with an error as low as 3%, which matches the accuracy levels recommended by relevant regulations.

Keywords: Flying robots · Mapping · Confined spaces · Autonomous.

1 Introduction

A confined space is an area that is substantially enclosed and where dangerous material or hazards inside the space or nearby may cause serious injury. One of the most challenging confined spaces for robotic inspection are cement manufacturing process storage spaces. This is due to robots having to operate under poor lighting and visibility conditions, lack of global positioning, uneven and slippery terrains, and sensor interference [1]. Drones have significant advantages over other mobile robotics. Hence, drones were used for inspecting confined spaces within many real-world applications such as sewer inspection [2], search and rescue (SAR) [3], and underground mines inspection and mapping [4], all

of which required an indoor localisation approach to deal with the limitations imposed by operation in confined spaces. However, the aforementioned examples are not suitable to export to the missions within cement plants storage because either they cannot perform within the exceptionally harsh environmental conditions or they are commercialised solutions that do not allow flexible adjustments to the system to fit new mission requirements.

One of the essential tasks conducted frequently in cement plants is to estimate the volume of different stockpiles within storage facilities. Estimating stockpile volumes in outdoor environments typically uses photogrammetry as it is cheaper and faster when compared to other methods like surveying [5]. Many studies, such as in [6], demonstrated real applications of using drones for stockpile volume estimation. Nevertheless, these studies have only focused on applying drones within outdoor environments. In fact, dust, limited illumination, and lack of GPS signals in confined spaces are some of the severe challenges that have rarely been considered in previous studies tackling aerial stockpile volume estimation.

Simultaneous Localisation and Mapping (SLAM) is one of the popular methods for drone navigation in GPS-denied environments. SLAM is the process of using cameras, Light Detection and Ranging (LiDAR), or both to estimate the robot's navigation states and the surrounding map simultaneously [7]. In dusty, low illuminated, and large confined spaces (such as in cement plants), LiDAR is the best class of sensors for such application [8]. Here, a scan-matching technique is usually employed to compute the current state transformation of the robot and compare it to last states [9]. Iterative Closest Point (ICP) algorithm is one of the most widely applied techniques for LiDAR scan-matching in the robotic community [10]. Since the introduction of ICP by Chen and Medioni [11] and Besl and McKay [12], many variants have been introduced to enable improvements such as reduction in overall computational cost, smaller mean square error, faster convergence speed, and optimal selection of points for overall algorithm efficiency. For more details on the different ICP variants, the reader is referred to Mora et al. [13] for a comprehensive review.

In this paper, we implemented a modified ICP algorithm for real-time scan-matching to localise a drone and simultaneously generate a 3D map of a confined space. The proposed method can be deployed in indoor, dark, and dusty facilities such as the storage spaces within cement plants. Matching high dense 3D scans provides detailed set of results that are needed for applications such as 3D reconstruction of power lines [14]. However, this is very costly to implement in real-time. Given that our intended application is to map a large stockpile that has no fine geometric details, we have the luxury/excuse to sacrifice some of the collected data without reducing the accuracy of the reconstructed map. As such, to reduce computational cost taken in matching scans and to reduce memory requirements, we reduced the gathered data rate from the 3D LiDAR through using the below horizon layers only, and applying the scan-matching at rates as low as every 6 seconds. Therefore, our proposed method obtained the 3D transformation matrix based only on the point clouds from the 2D horizontal layer. To increase the accuracy and speed of the matching, we assigned lower

weights to point clouds with greater distances from the LiDAR source because points at greater distances have more noise in dusty environments, as illustrated by Phillips et al. [8]. Whilst the model precisely simulates stockpiles in cement plants, it can be reconfigured to other applications such as agri-robotics, etc.

2 Method

2.1 Drone Localisation and 3D Map Generation

In a fully confined storage, the drone's position $(x_v, y_v, z_v, \phi, \theta, \psi)$ and the reconstructed 3D map $(\mathcal{M})$ of the scanned area are obtained by estimating the transformation matrices (rotation $\boldsymbol{R}_{3\times3}$ and translation $\boldsymbol{t}_{3\times1}$) between two LiDAR scans recorded at different locations using scan-matching based on point-to-point ICP algorithm. By maintaining flight at a constant height, h, we can simplify the problem and assume that $z_v = h$, $\phi = 0$ and $\theta = 0$. Therefore, the scan-matching process can be solved as a 2D problem by only employing the point cloud in the horizontal layer of the LiDAR scans to estimate the transformation matrices.

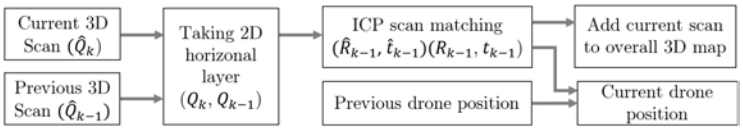

Fig. 1. Data flow diagram for 3D map reconstruction and drone's position estimating.

Figure 1 shows the data flow diagram for drone localisation and total map generation. Let $\hat{\boldsymbol{Q}}$ be a scan of size $3 \times n$ received from a 3D LiDAR that contains point clouds, i.e. each column (n) in $\hat{\boldsymbol{Q}}$ is a point cloud that is defined in the three-dimensional space by its coordinates (x, y, z). Moreover, let $\boldsymbol{Q}$ (size $2 \times n$) be the selected points by taking the horizontal layer from $\hat{\boldsymbol{Q}}$, so that each point cloud in $\boldsymbol{Q}$ is defined within the two-dimensional space by its coordinates (x, y). The applied ICP algorithm is shown in Algorithm 1. The required 2D rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$ can be estimated by minimising the sum of the squared error in:

$$E(\boldsymbol{R}, \boldsymbol{t}) = \sum_{i=1}^{N} w_i \left\| q_i - (\boldsymbol{R} p_i - \boldsymbol{t}) \right\|^2, \tag{1}$$

where q_i and p_i are the corresponding pair points from the current scan $\boldsymbol{Q}_k$ and the previous scan $\boldsymbol{Q}_{k-1}$, respectively. The subscript k and $k-1$ are the scan index, $\| \, . \, \|$ is the norm, N is the overall number of the matched point pairs, and w_i is a weighting factor for the ith pair. For every point q in $\boldsymbol{Q}_{k-1}$, a search is conducted for the corresponding point p in $\boldsymbol{Q}_k$ which has the closest distance. The weighting factor, w_i, is set to be linear from 1 to 0, where 1 is for points at minimum range of the LiDAR and 0 is for those at its maximum range.

Algorithm 1: Compute the 2D transformations ($\boldsymbol{R}_{k-1}$, $\boldsymbol{t}_{k-1}$) between two 3D LiDAR scans.

Input: Current 3D scan ($\hat{\boldsymbol{Q}}_k$) and previous 3D scan ($\hat{\boldsymbol{Q}}_{k-1}$)
Result: $\boldsymbol{R}_{k-1}$, $\boldsymbol{t}_{k-1}$
Initialisation: $\boldsymbol{Q} = f(\hat{\boldsymbol{Q}}) \leftarrow$ Take the 2D horizontal layer;
$E = \infty$;
$\boldsymbol{R} = [1\ 0; 0\ 1]$, $\boldsymbol{t} = [0\ 0]^T$;
while $E >$ *threshold, & iteration* $<$ *maximum number of iteration* **do**
$\boldsymbol{Q}_k = \boldsymbol{R}\,\boldsymbol{Q}_k - \boldsymbol{t} \leftarrow$ Update $\boldsymbol{Q}_k$;
Determine and weighting corresponding points $\boldsymbol{q}$ and $\boldsymbol{p}$;
Determine the mean of the corresponding,$\boldsymbol{u}_{Q'_i}$ and $\boldsymbol{u}_{Q'_{i-1}}$;
Compute the cross-coverance matrix, $\boldsymbol{K}$;
SVD($\boldsymbol{K}$) $\leftarrow \boldsymbol{U}, \boldsymbol{V}$;
$\boldsymbol{R}_{new} = \boldsymbol{U}\,\boldsymbol{V}^T \leftarrow$ New $\boldsymbol{R}$;
$\boldsymbol{R} = \boldsymbol{R}_{new}\,\boldsymbol{R} \leftarrow$ Update $\boldsymbol{R}$;
$\boldsymbol{t}_{new} = \boldsymbol{u}_{Q_{k-1}} - \boldsymbol{R}\,\boldsymbol{u}_{Q_k} \leftarrow$ New $\boldsymbol{t}$;
$\boldsymbol{t} = \boldsymbol{t}_{new} + \boldsymbol{t} \leftarrow$ Update $\boldsymbol{t}$;
$E(\boldsymbol{R}, \boldsymbol{t}) = \sum_{i=1}^{N} w_i \left\| q_i - (\boldsymbol{R} p_i - \boldsymbol{t}) \right\|^2$;
end

Algorithm 1 is executed for a specific number of iterations (maximum number of iteration), or until $E \leq$ *threshold*. After finding the transformation matrices, the drone current position can be obtained as:

$$\begin{bmatrix} x_v \\ y_v \end{bmatrix}_k = \boldsymbol{R}_{k-1} \begin{bmatrix} x_v \\ y_v \end{bmatrix}_{k-1} + \boldsymbol{t}_{k-1} \ , \tag{2}$$

where $\begin{bmatrix} x_v\ y_v \end{bmatrix}_{k-1}^T$ is the drone's previous position, and $\boldsymbol{R}_{k-1}$ and $\boldsymbol{t}_{k-1}$ are the obtained rotation and translation (transformation) matrices from Algorithm 1 that match the scan $\boldsymbol{Q}_k$ and $\boldsymbol{Q}_{k-1}$.

The overall 3D map, $\mathcal{M}$, of the total scanned area is generated by appending the current 3D scan $\hat{\boldsymbol{Q}}_k$ to the initial scan $\hat{\boldsymbol{Q}}_1$ after applying the obtained transformation matrices. Since the transformation matrices ($\boldsymbol{R}_{2\times2}$ and $\boldsymbol{t}_{2\times1}$) are cast in 2D space, we applied a homogeneous transformation to allow 3D space transformation matrices ($\hat{\boldsymbol{R}}_{\boldsymbol{3\times3}}$, $\hat{\boldsymbol{t}}_{\boldsymbol{3\times1}}$). The overall 3D map, $\mathcal{M}$, can, thus, be obtained as:

$$\mathcal{M}_{3\times n} = \begin{bmatrix} [\hat{\boldsymbol{Q}}_1]^T \\ [\hat{\boldsymbol{R}}_1\hat{\boldsymbol{Q}}_2 + \hat{\boldsymbol{t}}_1]^T \\ [\hat{\boldsymbol{R}}_2(\hat{\boldsymbol{R}}_2\hat{\boldsymbol{Q}}_3 + \hat{\boldsymbol{t}}_2) + \hat{\boldsymbol{t}}_1]^T \\ \vdots \\ [\hat{\boldsymbol{R}}_1(\hat{\boldsymbol{R}}_2(...(\hat{\boldsymbol{R}}_{k-1}\hat{\boldsymbol{Q}}_k + \hat{\boldsymbol{t}}_{k-1}) + ...) + \hat{\boldsymbol{t}}_2) + \hat{\boldsymbol{t}}_1]^T \end{bmatrix}^T = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \boldsymbol{z} \end{bmatrix}_{3\times n} , \tag{3}$$

where n is the total number of the registered point clouds and $\hat{\boldsymbol{Q}}_1$ is the initial 3D scan at scan index, $k = 1$.

2.2 Indoor Navigation

Navigating a drone within a dusty confined space is challenging due to the need to fly the drone beyond line of sight and the inability to use cameras [15]. To address the issue, a simple algorithm is implemented for autonomous navigation that always keeps the drone at a certain distance from the walls. The point clouds from the LiDAR horizontal layer are used to obtain the normal distance from the drone to the front (d_f) and right (d_r) obstacles, as illustrated in Figure 2. Let $\mathcal{R}_s$ be the ranges in the current scan $\boldsymbol{Q}_k$, where s is a positive integer that donates the range index. The distances d_f and d_r can be obtained from Equation (4) and (5) as follow:

$$d_f = \mathcal{R}_s, \quad \text{when } s = [\, r\pi/(2\pi)\,] = [\, r/2\,], \tag{4}$$

$$d_r = \mathcal{R}_s, \quad \text{when } s = [\, r(\pi/2)/(2\pi)\,] = [\, r/4\,], \tag{5}$$

where r is the total number of the ranges in $\mathcal{R}_s$ and $[\,.\,]$ denotes the standard rounding function that rounds to the closest integer number.

The navigation algorithm starts when the drone reaches the desired altitude, then the drone starts to fly forward whilst keeping $d_r = D$ by changing the heading direction using a proportional–derivative (PD) controller. Note that, D is the desired normal distance between the walls and the trajectory path. Then, when $d_f \leq D$, the drone starts turning. This way, the drone is always capable of keeping the distance, D, with the surrounding walls. Figure 2 shows an overall description of the trajectory planning and navigation parameters. Clearly, this is a very simple trajectory planning approach; however, it is sufficient for our storage mapping application.

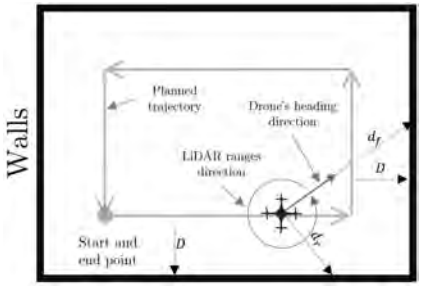

Fig. 2. The planned trajectory where d_f is the distance from the drone to the front obstacle and d_r is the distance to the right obstacle.

2.3 Surface Generation and Volume Estimation

At the end of the mapping process, the volume of the stockpile is estimated by calculating the volume of the overall 3D map, $\mathcal{M}$. The meshgrid function in Matlab is used to generate a uniform 2D grid across the inspected space $\mathcal{R}$. The heights, z, from $\mathcal{M}$ are interpolated on top of the uniform grid $\mathcal{R}$ using a

linear approach (achieved using the griddata function in Matlab). Therefore, the surface of the stockpile $Z_{surface}$ can be generated from these returned values. To estimate the volume of the stockpile, $V_{stockpile}$, double integration of the surface over the inspection space is, then, performed as follow:

$$V_{stockpile} = \iint_{\mathcal{R}} Z_{surface}\,(x, y)\,dx\,dy\ . \tag{6}$$

3 Simulation Setup

3.1 Simulation Environment

We created a model for the environment as well as the robotic system in Webots (2021a), an open-source virtual mobile robotics simulation platform that allows users to model, programme, and simulate mobile robots in a virtual prototyping environment. In Webots, a fully confined storage was implemented along with a stockpile as a fair representation of a real cement or clinker storage within a cement plant. The stockpile is a 3D CAD model of a generic stockpile designed in SolidWorks 2019 CAD software. Hence, the CAD software provides the actual volume of the stockpile that can be compared with the estimated volume. Figure 3 shows a screenshot of the modelled environment, including the implemented stockpile.

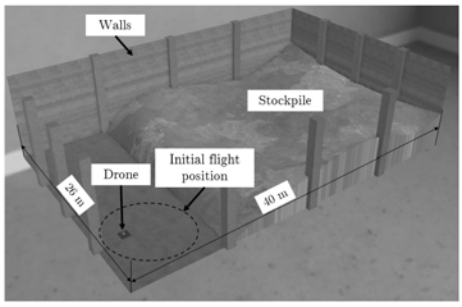

Fig. 3. A screenshot showing the developed model in Webots simulator demonstrating a stockpile in a fully confined storage. The ceiling and two walls are transparent to show the stockpile inside.

3.2 Robotic Platform

A quadcopter drone from the in-built robot libraries of Webots was inserted into the environment. To collect the 3D scans $\hat{\boldsymbol{Q}}$ of the environment, the drone was equipped with a rotary 3D LiDAR that has a 45°vertical field of view below the horizon. The LiDAR was set in the simulation with five layers. Each layer has 512 point clouds (i.e. low-dense scan). Moreover, a Gaussian noise with 0.12 standard deviation was added to the LiDAR data by the simulator. The drone was programmed and controlled using Matlab codes. Furthermore, the ICP scan-matching algorithm and the navigation approach were implemented in Matlab.

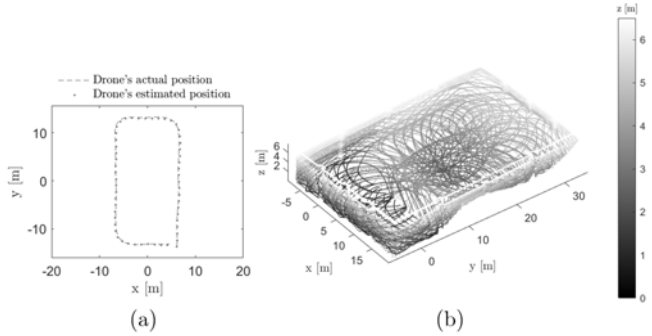

Fig. 4. Example simulation with $v_f = 0.43$ m/s and $s_r = 0.21$ Hz. (a) Estimated drone's positions superimposed on actual positions. (b) The overall 3D map ($\mathcal{M}$) of the total scanned area.

3.3 Simulations

To investigate the system's performance, we defined two parameters: drone forward speed, v_f, and scan rate, s_r. In Webots, a *pitch disturbance* value is typically used to move the drone forward. In this work, we tested three values of *pitch disturbance* $\in \{0.50, 0.75, 1.0\}$ which resulted in $v_f \in \{0.31, 0.43, 0.55\}$ m/s. Moreover, we recorded scans and applied the proposed scan-matching method described in Section 2 at four low-rate values by skipping $\{50, 100, 150, 200\}$ scans in the simulation loop which are equivalent to recording a scan every $\{1.6, 3.2, 4.8, 6.4\}$ seconds or $\{0.63, 0.31, 0.21, 0.16\}$ Hz. Each simulation was repeated five times with different initial positions. The doted circle in Figure 3 indicates the area where drone is randomly placed at the start of each test. Thus, in total, 60 tests were conducted. The threshold error and the maximum number of iteration were set in Algorithm 1 to 2 and 50, respectively. Lastly, the desired normal distance D between the walls and the trajectory path was set to 6 m.

In order to measure the performance of the system, three metrics were defined: i) computing cost, c_c, ii) error of the estimated positions, e_p, and iii) error of the estimated volume, e_v. The metric c_c is defined as the sum of total iterations in the ICP algorithm throughout each test. The metric e_p is defined as the root mean square error between the drone's actual and estimated positions. Finally, the metric e_v is defined as the percentage error between the actual and estimated stockpile volumes.

4 Results

Using the case with with $v_f = 0.43$ m/s and $s_r = 0.21$ Hz as an example, Figure 4-a shows the drone's estimated positions superimposed on the actual ones, whereas Figure 4-b shows the generated overall 3D map, $\mathcal{M}$. Evidently, the developed navigation strategy in Section 2.2 has successfully navigated the drone within the confined space by following the walls of the storage and successfully

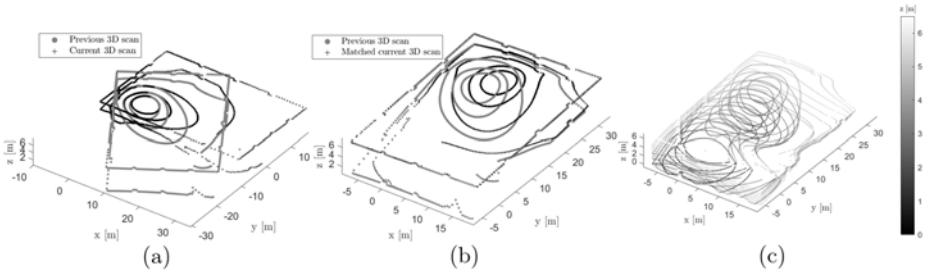

Fig. 5. An example of matching a current scan ($\boldsymbol{Q}_k$) with a previous scan ($\boldsymbol{Q}_{k-1}$) and adding the corrected current scan to the total map, $\mathcal{M}$. (a) Two scans before matching, (b) after matching, and (c) adding to $\mathcal{M}$. Colour in (c) encodes height from the ground.

returning to the initial position. Moreover, Figure 5 illustrates an example of matching a current scan ($\boldsymbol{Q}_k$) with a previous scan ($\boldsymbol{Q}_{k-1}$) and adding the corrected current scan to $\mathcal{M}$ in real-time. This matching would have not been achieved in real-time if all point clouds from both scans were used in the scan-matching. Therefore, our method and assumptions for matching 3D scans are very useful when scanning at a low-rate and whilst having low-dense scans.

The defined performance metrics of the system as well as the total recorded scans during simulation are assessed for the different flight speeds, v_f, and scan rates, s_r, Figure 6. It is evident that higher s_r and lower v_f increase all metrics c_c, e_p, and e_v. This is because as more scans are recorded (as seen in Figure 6-d) more scans have to be matched. Noting that the error E from Equation (1) at the end of each scan-matching has an influence on the next scan-matching due to the corrected current scan (with the error E) being a reference scan in the next scan-matching. Hence, more recorded scans will decrease the accuracy of the drone's position estimation and the reconstructed 3D map.

It should be noted that higher flight speeds (than the three speeds simulated) were not attempted to recognise safety considerations, as it is not safe to fly at high speeds within a confined space. On the other hand, lower scanning rate can lead to blind spots (un-scanned area) or failure of the scan-matching technique. As such, these considerations defined our maximum s_r value. It is evident from the shown results that the estimated volume is always more than the actual volume, mainly because of the walls and the ground that need to be excluded from $\mathcal{M}$. Nevertheless, the values of e_v are reasonable. In fact, according to [5], regulations regarding mine engineering often state that estimated volumes should present $\pm 3\%$ accuracy of the whole amount. hence, the proposed method for mapping succeeded to estimate the stockpile volume with the recommended accuracy.

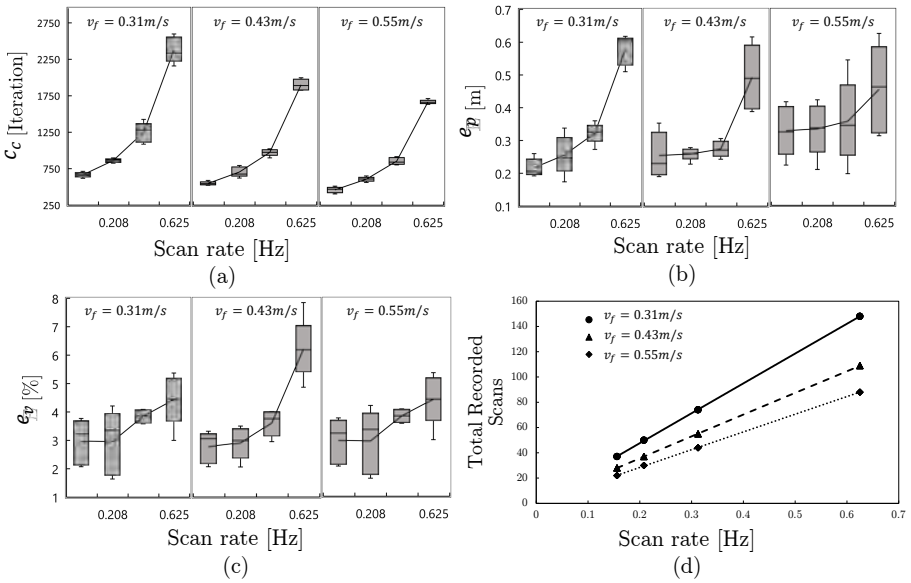

Fig. 6. Variations of the performance metrics of the system against drone forward speed, v_f, and scan rate, s_r. (a) Computing cost, c_c, (b) root mean square error of the estimated positions, e_p, (c) percentage error in volume estimation, e_v, and (d) the total recorded scans during the simulation. Each boxplot represents five repeated runs of each condition with different initial positions.

5 Conclusion & Future Work

This work demonstrated the implementation of a modified ICP algorithm for real-time scan-matching based on low-dense and low-rate scanning suitable for low computational requirements. The approach has localised a drone and simultaneously generated a 3D map of a confined space. A navigation strategy was developed to navigate the drone within the area autonomously. Using the reconstructed map of the confined space, the volume of the stored stockpile in the confined space was estimated.

The approach was tested using simulations for mapping a confined space within a cement plant. Results were demonstrated for different flight speeds and scan rates, demonstrating successful scan-matching. Moreover, the volume of the mapped stockpile was estimated with an error as low as 3%, which matches the accuracy levels recommended by relevant regulations.

In future work, we intend to develop the approach further by introducing a process to correctly remove outliers from the generated 3D map to enhance the accuracy of the results. Since the current method does not have a loop closure, our future work will, also, involve developing a strategy for accurate larger space mapping by closing the loop. We, then, intend to test the system in real-world scenarios.

References

1. A. Alsayed, M. R. Nabawy, A. Yunusa-Kaltungo, F. Arvin, and M. K. Quinn, "Towards Developing an Aerial Mapping System for Stockpile Volume Estimation in Cement Plants," in *AIAA Scitech 2021 Forum*, (Reston, Virginia), 2021.
2. C. H. Tan, M. Ng, D. S. B. Shaiful, S. K. H. Win, W. J. Ang, S. K. Yeung, H. B. Lim, M. N. Do, and S. Foong, "A smart unmanned aerial vehicle (UAV) based imaging system for inspection of deep hazardous tunnels," *Water Practice and Technology*, vol. 13, no. 4, pp. 991–1000, 2018.
3. D. Chatziparaschis, M. G. Lagoudakis, and P. Partsinevelos, "Aerial and ground robot collaboration for autonomous mapping in search and rescue missions," *Drones*, vol. 4, no. 4, pp. 1–24, 2020.
4. R. M. Turner, M. M. MacLaughlin, and S. R. Iverson, "Identifying and mapping potentially adverse discontinuities in underground excavations using thermal and multispectral UAV imagery," *Engineering Geology*, vol. 266, p. 105470, 2020.
5. P. Raeva, S. Filipova, and D. Filipov, "Volume computation of a stockpile-a study case comparing gps and uav measurements in an open pit quarry.," *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 41, 2016.
6. C. Arango and C. A. Morales, "Comparison between Multicopter UAV and Total Station for Estimating Stockpile Volumes," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XL-1/W4, no. 1W4, pp. 131–135, 2015.
7. M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, 2001.
8. T. G. Phillips, N. Guenther, and P. R. McAree, "When the dust settles: The four behaviors of lidar in the presence of fine airborne particulates," *Journal of field robotics*, vol. 34, no. 5, pp. 985–1009, 2017.
9. L. Shu, H. Xu, and M. Huang, "High-speed and accurate laser scan matching using classified features," in *IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, pp. 61–66, 2013.
10. X. Li, S. Du, G. Li, and H. Li, "Integrate point-cloud segmentation with 3d lidar scan-matching for mobile robot localization and mapping," *Sensors*, vol. 20, no. 1, p. 237, 2020.
11. Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image and vision computing*, vol. 10, no. 3, pp. 145–155, 1992.
12. P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611, pp. 586–606, 1992.
13. H. Mora, J. M. Mora-Pascual, A. García-García, and P. Martínez-González, "Computational analysis of distance operators for the iterative closest point algorithm," *PLOS ONE*, vol. 11, no. 10, pp. 1–19, 2016.
14. R. Zhang, B. Yang, W. Xiao, F. Liang, Y. Liu, and Z. Wang, "Automatic extraction of high-voltage power transmission objects from uav lidar point clouds," *Remote Sensing*, vol. 11, no. 22, p. 2600, 2019.
15. M. Petrlík, T. Báča, D. Heřt, M. Vrba, T. Krajník, and M. Saska, "A robust uav system for operations in a constrained environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2169–2176, 2020.