INDOOR MAPPING: EXPERIENCES WITH LIDAR SLAM

B. Suleymanoglu[1,2,*], M. Soycan[2], C. Toth[1]

[1]Department of Civil, Environmental and Geodetic Engineering, The Ohio State University,
470 Hitchcock Hall, 2070 Neil Ave., Columbus, OH 43210 - (Suleymanoglu.3, toth.2)@osu.edu

[2]Civil Engineering Faculty, Yildiz Technical University, 34220 Esenler Istanbul, Turkey – (bariss,soycan)@yildiz.edu.tr

Commission I, WG I/7

KEY WORDS: Indoor Mapping, UGV, LiDAR, SLAM, Sensor Fusion.

ABSTRACT:

Indoor mapping is gaining more interest in both research as well as in emerging applications. Building information systems (BIM) and indoor navigation are probably the driving force behind this trend. For accurate mapping, the platform trajectory reconstruction, or in other words sensor orientation, is essential to reduce or even eliminate for extensive ground control. Simultaneous localization and mapping (SLAM) is the computation problem of how to simultaneously estimate the platform/sensor trajectory while reconstructing the object space; usually, a real-time operation is assumed. Here we investigate the performance of two LiDAR SLAM tools based on using indoor data, acquired by a remotely controlled robot sensor platform. All comparisons were performed on similar datasets using appropriate metrics and encouraging results were obtained as a consequence of initial test studies yet further research is needed to analyse these tools and their accuracy comprehensively

1. INTRODUCTION

To map unknown environments in real-time using different kinds of sensors while estimating the location of mobile robotic systems at the same time is known as Simultaneous Localization and Mapping (SLAM) (Leonard and Durrant-Whyte, 1991). In the past two decades, research and development of navigation system for autonomous robots and driverless vehicles have been at the forefront of countless research studies, where SLAM algorithms play a significant role. The most important requirement for the navigation of robots and autonomous vehicles is a good knowledge of their own location and a good perception of their surroundings (Kong and Lu, 2017). Today, GNSS systems are used as the most common positioning system, as they provide precise positioning opportunities for different user needs. Of course, these systems have some drawbacks depending on the environment, such as tunnels, indoors, and urban canyons (Lin et al., 2018) not mentioning intentional or unintentional RF interference. Therefore, as an alternative, SLAM methods are one of the most widely used techniques for accurately estimating poses and trajectories in relative sense and perceiving the environment in order to provide navigation for self-driving vehicles and mobile robots in different dynamic outdoors and indoor environments where there are many obstacles (Bresson et al., 2017).

In order to solve the localization problem of driverless vehicles and mobile robots, different sensors can be used, such as LiDAR, RADAR, and monocular/stereo cameras. However, in general, camera and LiDAR are the most widely used sensors for this purpose. While a significant part of the SLAM research was devoted to passive visual sensors, LiDAR SLAM solution have received less attention, though LiDAR sensors have important practical advantages compared to passive cameras (Cadena et al., 2017). LiDAR systems, which are active sensors, can work independently of the light situation, perform better obstacle detection and tracking, and are increasingly becoming the basic measurement and positioning sensor for robotic studies (Cwian et al., 2021). For this reason, studies on the development of SLAM algorithms for LiDAR data and the examination of the performance of the developed methods are a hot topic among the scientific community. In this context, many LiDAR-based SLAM solution have been proposed, i.e., Cartographer (Hess et al., 2016), LOAM (Zhang and Singh, 2014), Hector SLAM (Kohlbrecher et al., 2011), which nearly all of them implemented in robot operating system (ROS) and numerous studies comparing these algorithms are available in the literature (Filipenko and Afanasyev 2018, Zou et al., 2020). However, due to the complexity of the ROS program architecture, using and testing these platform-based methods require a certain amount of experience and expertise. This can create difficulties for users with insufficient experience (Mihalik et al. 2021). In addition, among the many SLAM algorithms developed for LiDAR data, graph-based approaches are generally used in 3D LiDAR SLAM studies, and apart from the ROS environment, there are different LiDAR-based SLAM algorithms and software implementations. (Li et al., 2020). Therefore, within the scope of this study, it is aimed to compare LiDAR-based SLAM methods by using different software opportunities such as Matlab and LidarView and to contribute to the literature on this subject.

In this study, the performance of estimating mobile robot trajectories computed from 3D LiDAR-based graph SLAM using 3D LiDAR point cloud data is investigated. For this purpose, a mobile robot consisting of Velodyne VLP-16 LiDAR and different camera sensors was used. The data acquisition was conducted along a corridor on the second floor of the Bolz Hall at OSU. In post-processing mode, the trajectory of the mobile robot was estimated by using a Graph-based SLAM framework using the Matlab software and LidarView based SLAM technique. Several data acquisition sessions were done in the

[*] Corresponding author

study area and the results of the LiDAR SLAM were compared with ground truth. Note that reference measurements were obtained in some parts of the study area during the analysis, trajectory error and error rate were calculated. In addition, the results obtained with both software are examined in detail and their relative accuracy is estimated.

2. LIDAR SLAM TECHNIQUES

2.1 LidarView SLAM Algorithm

To support this study, an open source version of the LidarView software developed by Kitware was used. This tool provides a large number of algorithms and features for visualization, processing, evaluation and recording of point cloud data. More importantly, LidarView has an embedded SLAM algorithm solution to process raw 3D LiDAR data. This algorithm was developed with inspiration from the LOAM method and can be adapted to LiDAR data obtained from different environments by changing parameter settings. Detailed information about how to SLAM with LidarView are available on the website given in reference section (LidarView SLAM, 2021)

2.2 Matlab SLAM for 3D LiDAR Point Clouds

The functions included in the Matlab software allow the implementation of 3D point cloud based SLAM. The 3D LiDAR SLAM method uses many libraries from the navigation and computer toolbox. The SLAM method implemented in Matlab consists of pre-processing point clouds, registering point clouds, detect loops, correcting drift, creating maps and localizing steps, Fig. 1 (Matlab, 2022). In order to prepare the point cloud data for the registration process, the first pre-processing step is to reduce the number of points by filtering unwanted points, such as some object features that represent noise within a dataset from our perspective. In the next step, the registration process is carried out in order to estimate the trajectory. In this context, the Normal Distribution Transform (NDT) registration function was used (Biber and Straber, 2003). Note that Matlab allows for a different number of registration functions to be used for SLAM computation. In this context, these functions can be selected depending on the nature of the test area for various research purposes. In the final step, pose graph optimization technique is used to detect loop closures and correct drifts.

3. EXPERIMENTS AND DISCUSSION

In this section, the evaluations of the results obtained by 3D LiDAR SLAM-based methods, embedded in Matlab and LidarView software are explained in detail. All data used in the study were collected with our own design unmanned ground vehicle (UGV) system in indoor environments.

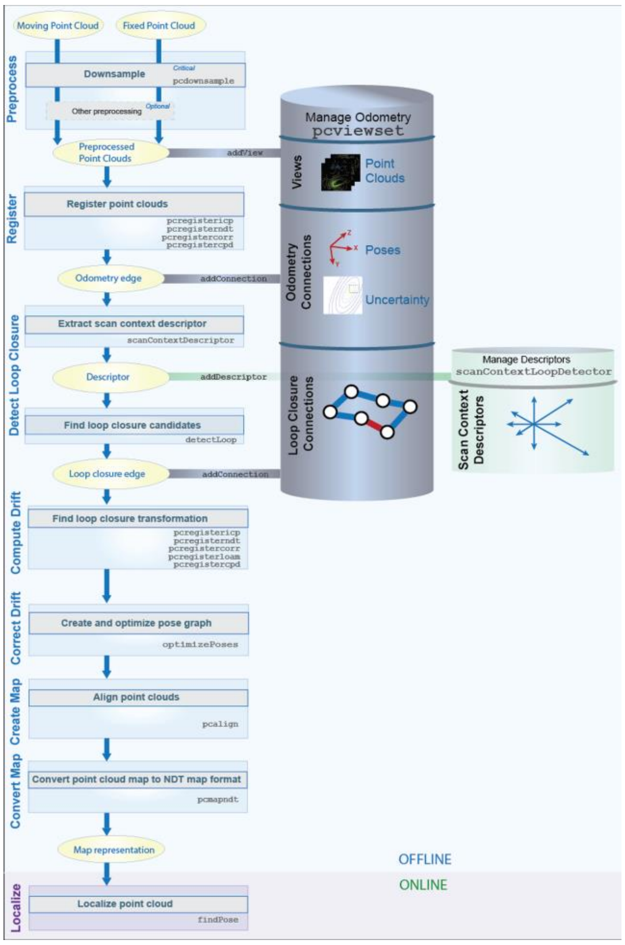


Figure 1. General workflow steps for Matlab LiDAR SLAM (Matlab 2022)

3.1 System Configuration and Experimental Environment

In this study, based on a Loomo system (Loomo, 2021), we prototyped a human operated UGV for our experiments. Loomo developed by Segway is a self-balancing vehicle (SBV) that includes many capabilities, such as visual perception and understanding, supported by computer vision and deep learning technologies. Fig. 2 presents system configuration of our UGV. This platform carries different types of sensors, such as Velodyne VLP-16 LiDAR, GoPro HERO5 Black, Casio EX-H20G and Microsoft Kinect XBOX 360 camera. A laptop is used for recording Microsoft Kinect 360 data, Raspberry Pi is used to record LiDAR data and the all these systems are powered by a main power supply. For easier handling, the supporting hardware, including the laptop, Raspberry Pi, and power supply are placed on a pushcart next to the UGV and an operator drives the cart following the robot during data acquisition. In this study, the UGV was controlled remotely with the help of a mobile application which provided a waypoint list for a specific trajectory. During the acquisition, all the sensor data were recorded, though not all the sensor data streams are considered in this effort. In several sessions, the robot was moving in the pre-planned trajectory patterns.

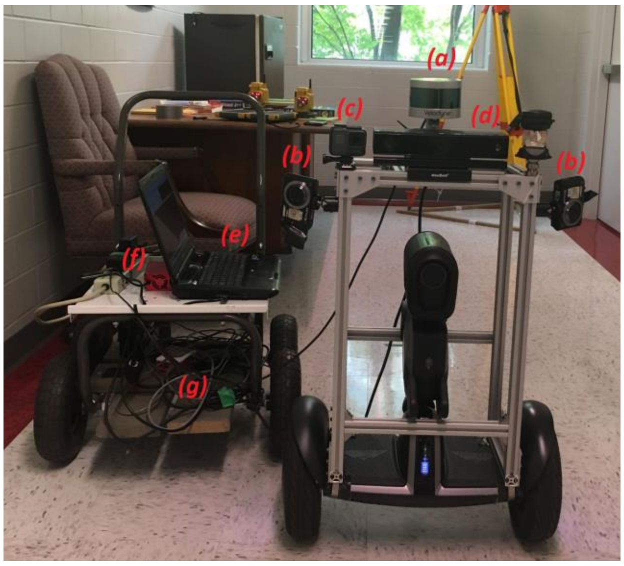


Figure 2. Sensors: a) Velodyne VLP-16 LiDAR, b) Casio, c) GoPro, d) Microsoft XBOX Kinect 360 Sensor; Hardware: e) Laptop, g) Raspberry pi, (f) Power Source

For the purpose of this study, only the data obtained from the LiDAR sensor was used. The Velodyne VLP-16 LiDAR supports 16 channels and has a maximum measurement range of 100 m with a horizontal field of view 360^0, with a skewed vertical field of view 40^0, with an accuracy of ± 3 cm and angular resolution of (horizontal/azimuth) 0.1 - 0.4 (Velodyne, 2022). While it is not recent sensor, the Velodyne VLP-16 is adequate for the needs of this study. Information about UGV hardware and software system configuration is provided in Table 1.

As indicated above, multiple data acquisition sessions were conducted along a corridor on the second floor of the Bolz Hall at OSU; an area of approximate size of 38m x 63m. This study area is large enough to investigate LiDAR SLAM techniques and yet provides a moderately challenging environment with few objects on both sides of the corridors. The ground truth of UGV trajectories was tracked by a robotic total station as well as measured manually. Within test area, the UGV follows L-shape (A-B-C) and rectangle shape (C-D-E-F-G-C) trajectories as shown in Fig. 3. In addition, though this study focused on trajectory extraction, yet a loop closure was performed.

Platform	Loomo			
LiDAR Velodyne VLP-16	H/V: 1.33º/0.2º	Front bottom	20 Hz	H/V: 30º, 360º
Casio EX-H20g	1280 x 720	Back right Back left	30 Hz	72º x 57º
GoPro HERO5 Black	3840 x 2160	Front right	30 Hz	102º x 70º

Table 1. Sensor specification of the UGV system

3.2 Accuracy Evaluation

The accuracy assessment of the LiDAR SLAM techniques tested in this study was based on using a similar methodology proposed in (Zou et al., 2020). The evaluation was carried out based on the different test runs with the UGV system for the A-B (straight line), A-B-C (L shape) trajectories. These points were marked manually on the floor and the distances between them were measured and determined as A-B = 40.5 m, B-C = 9.5 m. For each trajectory, we considered the starting point as the origin of a local coordinate system. Since the results of the trajectory outputs of SLAM algorithms are independent and different in data rate, the positioning error was not analysed on a point basis. Instead, error (σ) and error rate (φ) values are estimated for the various trajectory locations.

$$\sigma_i = \sqrt{\left(x_i - x_j\right)^2 + (y_i - y_j)^2} \tag{1}$$

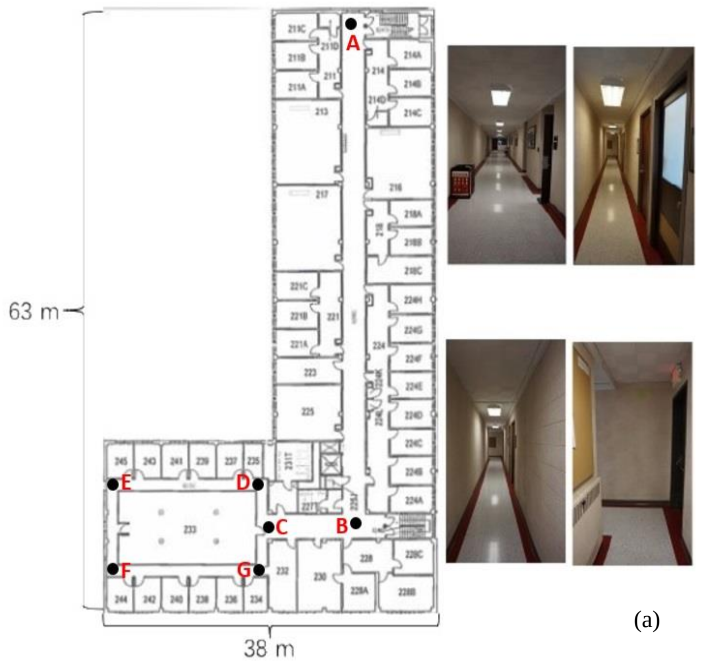
$$\varphi = \frac{\sigma}{L} \times 100\% \tag{2}$$

Where σ is the error between generated trajectory position and reference position

x_i, y_i = reference trajectory position
x_j, y_j = LiDAR SLAM trajectory position
L = total length of test area (Zou et al., 2020)

However, since the reference measurement were not yet available in the C-D-E-F-G corridor segment of the study area, the results here were compared visually and the relative differences of the segments were examined.

In addition to these analyses, some key places were selected along the A-B corridor segment for the detailed analysis of the localization performances of the LiDAR SLAM software used in our study. In this context, the width and height of the corridor, the width and depth of the door entrances, and the distances between the door edges were selected as key points and were manually measured to evaluate accuracy performance. Figure 3(b) illustrates selected key features for evaluating localization accuracy. Accordingly, 6 door distances, 6 door entrance features, and corridor width and height were measured and compared to SLAM-based position.



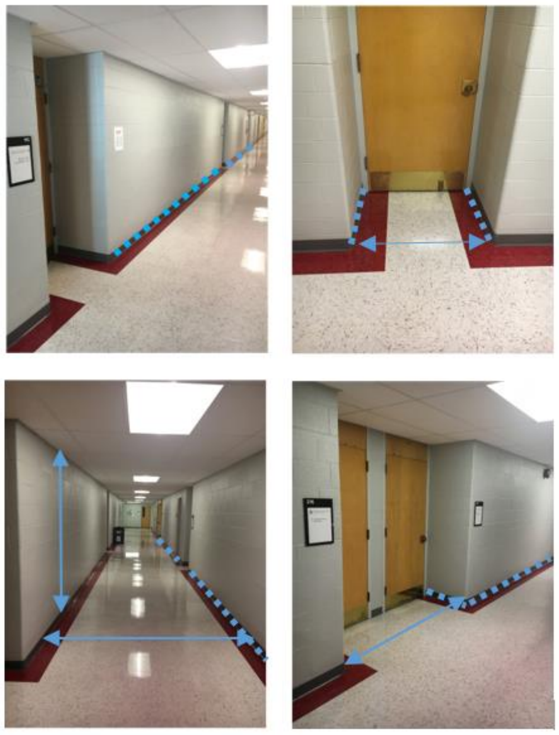


Figure 3. Layout for test corridor and image samples for these corridors (a) (Yuan, 2021), Key features selected for localization performance (arrows indicate corridor width and height and width of door entrances, dashed lines indicate depth of the door entrances and distances between door edges.) (b)

.

3.3 Results

In Fig. 4 and Table 2, the SLAM results obtained with LidarView and Matlab are shown, both visually and numerically. Looking at the trajectories, it is seen that both methods produce relatively good results along the longer corridor line (A-B) and the shorter B-C line. When the two tools are compared based on their internal accuracy estimates, then it can be stated that the Matlab software has slightly lower accuracy than LidarView for B-C corridor, while higher accuracy is achieved for A-B corridor. Despite the fairly low error rates and visually both software show potential and is expected to be improved with proper configuration. In general, the corridor lines are detected well, but clearly, there is a scale problem in terms of length estimations in both software as illustrated in figure 4. In this context, more test studies are needed to determine the best parameter settings for these software tools.

Trajectory (A-B)	**Error (m)**	**Error Rate**
Matlab	0.4	0.9 %
LidarView	0.85	2 %
Trajectory (B-C)		
Matlab	0.2	2.1 %
LidarView	0.15	1.6 %

Table 2. Error values for both trajectories

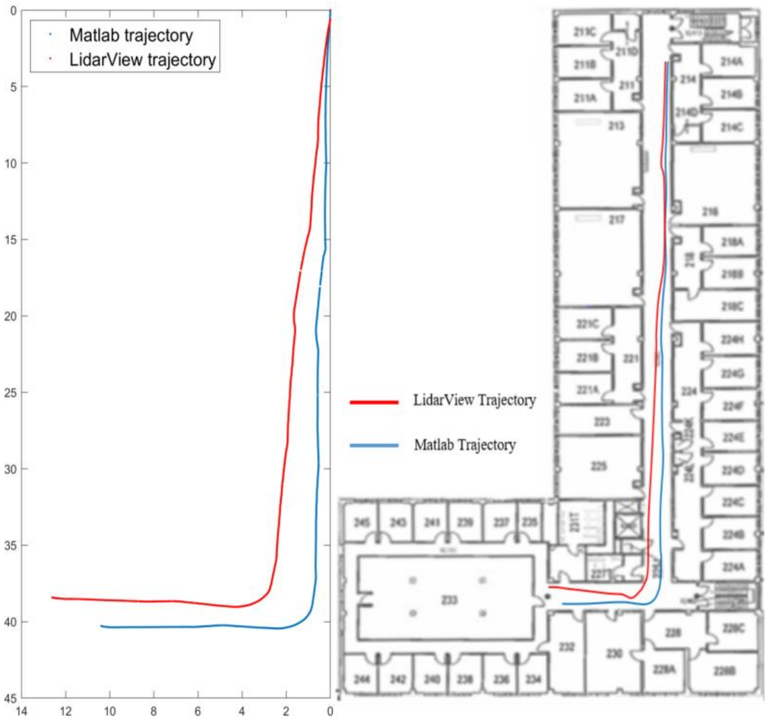


Figure 4. Trajectory for A-B-C corridor section, Blue line: Matlab trajectory, Red Line: LidarView (Left Side), Trajectory data matched with the underlying floor plan (Right Side)

The results obtained for the C-E-F-G corridor segment were examined in relative terms as stated earlier. The trajectories obtained by both software, shown in Fig. 5 are very similar to each other in terms of shape, but there is a difference in rotation. There is a small noticeable difference in the two trajectories, favouring the shape obtained by Matlab, which is visually better than LidarView in terms of consistency. In addition, trajectory length obtained in Matlab and LidarView for each segment were calculated and the differences between them were examined, see Table 3, which shows that the differences between the distances obtained from both systems are not to big, and in fact, both systems were able to effectively extract the trajectory in this narrow and long corridor.

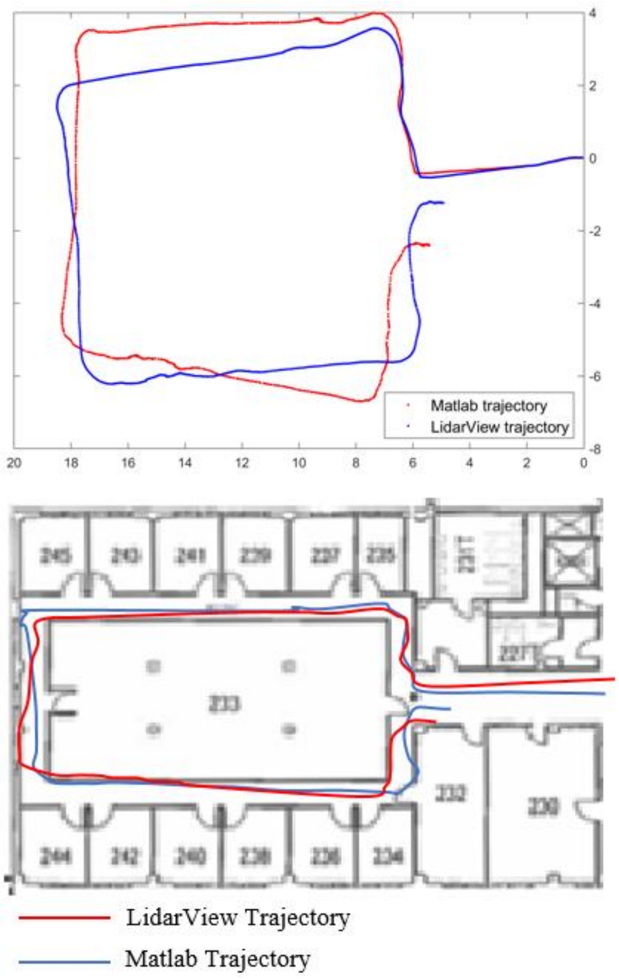


Figure 5. Trajectory for C-D-E-F-G corridor section, Blue line: Matlab trajectory, Red Line: LidarView (Top View), Trajectory data matched with the underlying floor plan (Bottom View)

Trajectory (m)	Matlab	LidarView	Difference (m)
C-D	3.3	3.3	0
D-E	11.2	10.6	0.6
E-F	7.8	8.4	0.6
F-G	11.3	10.7	0.6

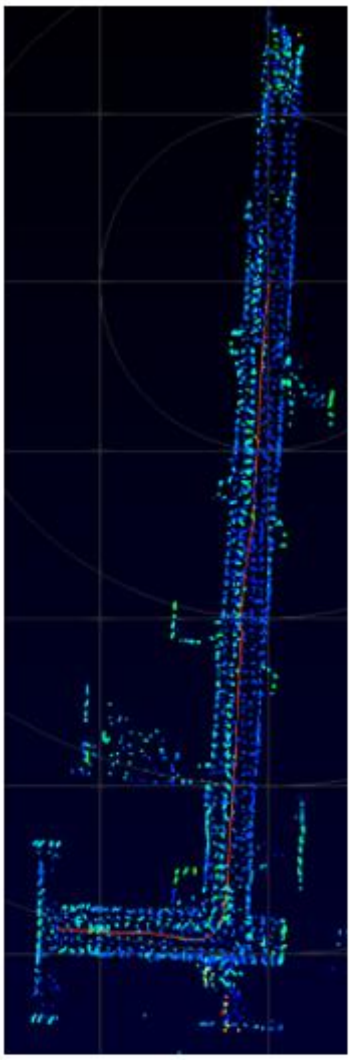
Table 3. Trajectory lengths obtained from both software for C-D-E-F corridor segment

Furthermore, as stated in the previous section, some key features were selected and measured in the A-B corridor section to evaluate the localization accuracy of both software. These key features include the distances between the door corners, the width and height of the corridor, and the dimensions of the doorway, as illustrated in figure 3(b). Table 4 shows the relative positions of these features and localization errors obtained from both software. Since the corridor width and height were measured more than once at different locations of the SLAM maps, they were averaged. Similarly, the averages of the dimensions of a large number of similar door entrances were taken. When the results were examined, it was seen that the distances obtained from both systems were close to each other, and the average localization error was found to be 0.24 m for Matlab-SLAM and 0.3 m for LidarView. This has shown that both systems produce results that are consistent with the environment.

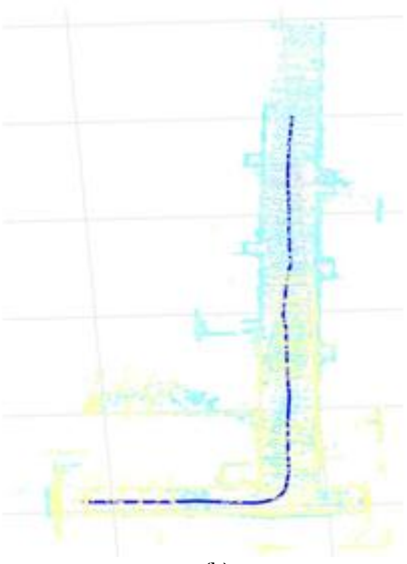
		Ground Truth	Matlab SLAM	LidarView SLAM
DDE1 (Distance between doors)		7.90 m	7.75 m	7.85m
DDE2		8.06 m	7.6 m	7.55m
DDE3		8.29 m	8.05 m	7.76m
DDE4		13.94 m	13.60 m	13.42m
DDE5		8.35 m	7.62 m	7.72m
DDE6		7.89 m	7.33 m	7.55 m
			Average	Average
Corridor Width		2.54 m	2.70 m	2.60 m
Corridor Height		2.38 m	2.40 m	2.45 m
Door Entrance	Width	1.02 m	1 m	0.90 m
	Depth	0.76 m	0.71 m	0.72 m

Table 4. Comparison of the localization error for different key features

Fig. 6a below shows 3D point cloud data generated by LidarView SLAM with red dots marking trajectory. Fig. 6b depicts point cloud obtained from Matlab SLAM with blue dots marking trajectory. Comparing the figures, the LidarView generated data appear to be smooth and stable for the L-shape corridor section. In contrast, the map obtained with Matlab SLAM is not as smooth and consistent as that of the LidarView, and in fact, it shows noticeable amount of distortion. Obviously, trajectories and point clouds are consistent within each software.



(a)



(b)

Figure 6. 3D Point Cloud Map; a) LidarView (Red dots trajectory) and b) Matlab (Blue dots trajectory)

4. CONCLUSION

In this study, the performance of two different SLAM algorithms based on Matlab and LidarView has been evaluated based on experimental indoor data. The initial results show higher error rates than expected; note that most of the similar studies come from the robotics community, using systems built around ROS environment. The visual evaluation showed moderate and thus encouraging performance. Clearly, more investigations are needed, including additional sensors and test areas as well as various methods in multiple configurations.

ACKNOWLEDGEMENTS

The study was sponsored by The Scientific and Technological Research Council of Turkey (TUBITAK) with grant number 1059B141900561.

REFERENCES

Biber, P., Straßer, W. 2003. The normal distributions transform: A new approach to laser scan matching. In Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453) (Vol. 3) 2743-2748.

Bresson, G., Alsayed, Z., Yu, L., Glaser, S., 2017. Simultaneous localization and mapping: A survey of current trends in autonomous driving. IEEE Transactions on Intelligent Vehicles 2(3), 194-220.

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Leonard, J. J., 2016. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. IEEE Transactions on robotics, 32(6), 1309-1332.

Ćwian, K., Nowicki, M. R., Wietrzykowski, J., Skrzypczyński, P., 2021. Large-Scale LiDAR SLAM with Factor Graph Optimization on High-Level Geometric Features. Sensors, 21(10), 3445.

Filipenko, M., Afanasyev, I. 2018. Comparison of various slam systems for mobile robot in an indoor environment. In 2018 International Conference on Intelligent Systems (IS) 400-407. doi: 10.1109/IS.2018.8710464

Hess, W., Kohler, D., Rapp, H., Andor, D. 2016. Real-time loop closure in 2D LIDAR SLAM. IEEE International Conference on Robotics and Automation (ICRA) 1271-1278. doi: 10.1109/ICRA.2016.7487258

Mihálik, M., Hruboš, M., Janota, A., 2021. Testing of SLAM methods in the Matlab environment. *IEEE 19th World Symposium on Applied Machine Intelligence and Informatics*. 55-58.

Kohlbrecher, S., Von Stryk, O., Meyer, J., Klingauf, U., 2011. A flexible and scalable SLAM system with full 3D motion estimation. In 2011 IEEE international symposium on safety, security, and rescue robotics 155-160. doi: 10.1109/SSRR.2011.6106777

Kong, Z., Lu, Q. 2017. A brief review of simultaneous localization and mapping. In IECON 2017-43rd Annual

Conference of the IEEE Industrial Electronics Society 5517-5522. doi:10.1109/IECON.2017.8216955

Leonard, J. J., Durrant-Whyte, H. F., 1991. Simultaneous map building and localization for an autonomous mobile robot. In IROS Vol. (3), 1442-1447.

Li, X., Du, S., Li, G., Li, H., 2020. Integrate point-cloud segmentation with 3D lidar scan-matching for mobile robot localization and mapping. Sensors 20(1), 237.

LiDARView, http://lidarview.com/

LidarView SLAM, 2021. Using SLAM in LidarView https://gitlab.kitware.com/keu-computervision/slam/-/blob/master/paraview_wrapping/Plugin/doc/How_to_SLAM_with_LidarView.md#installing-lidarview-or-one-of-its-derivative-with-slam-support (5 January 2022)

Lin, Y., Gao, F., Qin, T., Gao, W., Liu, T., Wu, W., Shen, S. 2018. Autonomous aerial navigation using monocular visual-inertial fusion. Journal of Field Robotics 35(1), 23-51.

Loomo by Segway (2022), https://store.segway.com/segway-loomo-mini-transporter-robot-sidekick

Matlab SLAM, 2022. Implement simultaneous localization and mapping (SLAM) https://www.mathworks.com/help/nav/ug/implement-simultaneous-localization-and-mapping-with-lidar-scans.html (15 January 2022)

Velodyne (2022) https://velodynelidar.com/products/puck/

Yang, Y. 2021. Indoor Positioning System for Smart Devices PhD dissertation, The Ohio State University

Zhang, J., Singh, S., 2014, July. LOAM: Lidar Odometry and Mapping in Real-time. In Robotics: Science and Systems 2(9).

Zou, Q., Sun, Q., Chen, L., Nie, B., Li, Q., 2021. A Comparative Analysis of LiDAR SLAM-Based Indoor Navigation for Autonomous Vehicles. IEEE Transactions on Intelligent Transportation Systems.