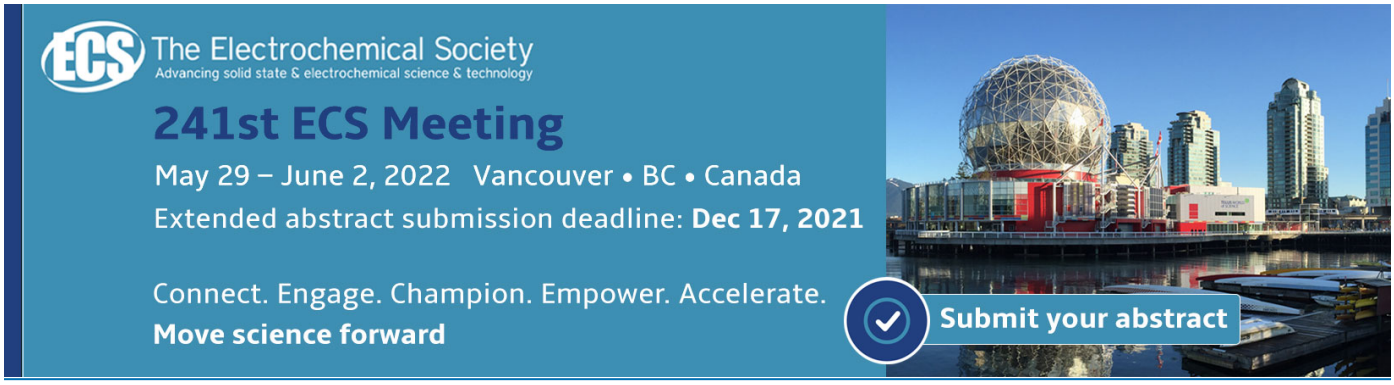
# 2-D UAV navigation solution with LIDAR sensor under GPS-denied environment





# 2-D UAV navigation solution with LIDAR sensor under GPS-denied environment

**J C Ho, S K Phang* and H K Mun**

School of Computer Science and Engineering, Taylor's University, No.1 Jalan Taylor's 47500 Subang Jaya, Selangor.

*E-mail: sweeking.phang@taylors.edu.my

**Abstract.** Unmanned aerial vehicle (UAV) is widely used by many industries these days such as militaries, agriculture, and surveillance. However, one of the main challenges of UAV is navigating through an environment where global positioning system (GPS) is being denied. The main purpose of this paper is to find a solution for UAV to be able to navigate in a GPS denied surrounding without affecting the drone flight performance. There are two ways to overcome these challenges such as using visual odometry (VO) or by using simultaneous localization and mapping (SLAM). However, VO has a drawback because camera sensors require good lighting which will affect the performance of the UAV when it is navigating through a low light intensity environment. Hence, in this paper 2-D SLAM will be use as a solution to help UAV to navigate under a GPS-denied environment with the help of a light detection and ranging (LIDAR) sensor which known as a LIDAR-based SLAM. This is because SLAM can help UAVs to localize itself and map the surrounding of the environment. The concept and idea of this paper will be fully simulated using MATLAB, where the drone navigation will be simulated in MATLAB to extract LIDAR data and to use the LIDAR data to carry out SLAM via pose graph optimization. Besides, the contribution to this research work has also identified that in pose graph optimization, the loop closure threshold and loop closure radius play an important role. The loop closure threshold can affect the accuracy of the trajectory of the drone and the accuracy of mapping the environment as compared to ground truth. On the other hand, the loop closure search radius can increase the processing speed of obtaining the data via pose graph optimization. The main contribution to this research work is shown that the processing speed can increase up to 45 % and the accuracy of the trajectory of the drone and the mapped surrounding is quite accurate as compared to ground truth.

## 1. Introduction

Unmanned Aerial Vehicle (UAV) which is commonly known as drone is a type of aircraft that is being controlled remotely or by onboard computers. During the early stage, when UAVs were just being developed, it is commonly used by militaries as a target decoy and to carry out missions during combat. However in this era of industry 4.0, UAVs are applied in many applications by different industries such as tracking and landing on moving ground vehicle [1]. UAVs are usually controlled by human from a ground station during the early stage of drone development. However, due to the improved in technologies and software, UAVs are able to fly autonomously via a path that was predetermined and programmed called waypoints [2]. This could be done with the help of Inertial Measurement Unit (IMU) which consists of accelerometers and gyroscopes that can aid in finding the linear and angular movement of the drone from the initial point of flight. Nevertheless, IMU will produce a small error due to drone drifting. Hence, a Global Positioning System (GPS) is also needed

to help recover the error that was suffered by the IMU as it adds up overtime [3]. Therefore, GPS plays a big role in unmanned vehicle as the position of the vehicle is important for certain activies such as path tracking performance and path planning [4].. Therefore, this work is to introduce a 2-D navigation system for UAVs to navigate in a GPS denied environment by using Light Detection and Ranging (LIDAR) sensors and a programmed called Simultaneous Localization and Mapping (SLAM). SLAM is actually very important as it helps to provide data for certain application such as path planning via Obstacle Avoidance and Navigation (OAN) algorithm [5].

The reason that why SLAM is the main solution to be used in UAV navigation in a GPS denied environment is because SLAM algorithms enables UAV to self-localize itself in the environment and at the same time it is able to map the environment to identify the obstacle and the path that is available in the environment [6]. Hence, although there are no GPS signals available, the UAV are able to navigate within the free spaces safely and at the same time allows the drone to explore the environment [7]. There are two types of SLAM, the first SLAM is called visual SLAM (vSLAM) which uses RGB-D camera, monocular vision camera and LIDAR SLAM which uses LIDAR sensors [8]. Researches has tested out a UAV navigation solution using monocular camera and optical sensors and it is proven that both this camera based SLAM is efficient in navigating the drone in a partially or unknown environment [9][10][11]. However, the drawback of using camera-based sensors is that it requires good lighting to obtain a more accurate data. Besides, a research was done by Fei wang et.al, which uses two laser range sensors to obtained data in a 3-D platform to help UAV to navigate in a partially known environment [12] but the process is complicated as it requires to examine in the x, y and z direction.

In this paper we will be mainly focusing on LIDAR SLAM instead of vSLAM because LIDAR provides a more accurate and precise data on the environment. Therefore, LIDAR sensor will be used in this paper to generate LIDAR DATA for the drone to carry out SLAM via pose graph optimization. Besides, a further contribution will also be done to understand the pose graph optimization method and to increase the processing speed of pose graph optimization. The remainder of this paper will be separated into three sections, whereby the first section is to introduce on the fundamentals on how localisation and mapping is carried out through pose graph optimization, follow by the methods of approach on how LIDAR data is generated and how the LIDAR data is being implemented to carry out 2-D LIDAR SLAM, the key results of the implemented codes and lastly ending of with a conclusion and final thoughts on the proposed work.

### *1.1 Pose Graph Optimization*

Pose Graph Optimization is considered a graph-based method which formulate SLAM so that the underlying spatial structure can be identified to determine the pose and map the surrounding of the environment simultaneously. Pose graph-based dimensions uses a method called least squares as an approach for computing a solution of an overdetermined system which have more equations and more unknowns. In a pose graph optimization, it mainly consists of nodes and edges. Every node in this graph represents the pose of the robot at a certain point. On the other hand, we have the edges, and these edges are kind of the relation between those poses or nodes which contains the constraint that relates between any two robot poses and these constraints could be obtain by the LIDAR data measurements through scan matching or feature matching which is done in the front-end section as shown in Figure 1.

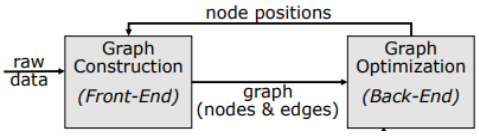



**Figure 1.** Concept and fundamental of pose graph optimization

A pose graph optimization consists of a front end where it is constructing the graph and a back end which is optimizing the graph. In the front-end part raw data is being accepted and it is used as an observation which then is converted into constraints through methods such as feature matching, feature extraction and pose estimation. These constraints that was created will then be used at the back-end part where the constraints are being used to optimize the graph in returning the corrected poses. This is an interplay between the front end and the back end where it relates each other by creating constrains and this constrains will then be optimized and the optimized map is used to make new data stations. During graph construction the LIDAR data that was generated will be used to form the poses as well as the constraints through a method call feature extraction, feature matching, and pose estimation. At the first stage the graph that was obtain will consist of n nodes x whereby,

$$x = x_{i:n} \tag{1}$$

Based on Equation 1 it means that the overall graph represents all end nodes known as x which consists of all the nodes put together with the unknown $x_i$ which is the pose of the robot at the given time $t_i$. An edge exists when a node $x_i$ moves to a position known as $x_{i+1}$. Which means its move from one point to another point and this edge was created by the first stage which is during feature construction.

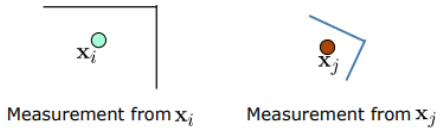



**Figure 2.** Formation of the edge through loop closure constraints

An edge can also exist between two arbitrary poses so that $x_i$ and $x_{i+1}$ $(x_j)$, where $x_j$, is known as the nodes where it contains similar data with $x_j$ as shown in Figure 2. Which means that at the point $x_i$ the robot is currently observing this laser scanner and at the point $x_j$it is also seeing the same data as $x_i$ and when relating the data of $x_i$ and $x_j$ we could form an observation known as the virtual observation. Which is give an idea of where $x_j$ should be as seen from $x_i$ which is known as a virtual measurement.

The third stage is known as transformation. In this stage a homogenous coordinate is being used whereby. Equation 2 shows the odometry-based edge whereby, ${x_i}^{-1}$ is the translation of $x_{i+1}$ and Equation 3 shows the observation-based edge whereby, ${x_i}^{-1}$ is the translation of $x_j$

$${x_i}^{-1}x_{i+1} \tag{2}$$

$${x_i}^{-1}x_j \tag{3}$$

${x_i}^{-1}$, $x_{i+1}$ and $x_j$ are known as the homogeneous matrixes. These homogeneous matrixes are used for translation and rotation of coordinates in projective geometry to identify the transformation of the poses.

Lastly, to generate the pose graph, the data obtained has to be optimized to reduce the errors that was form between the nodes. This is because, by reducing the errors known as $\Omega_{ij}$ between the nodes that has the constrains we can optimize the data to form the actual data by reducing the amount of error that is form between those two nodes. This can be done by using a method known as the least squares approach whereby,

$$x* = argmin \sum_{ij} e_{ij}^{T}(x_i, x_j)\Omega_{ij} e_{ij}(x_i, x_j) \quad (4)$$

$$x* = argmin \sum_{ij} e_{ij}^{T}(x)\Omega_k e_k(x) \quad (5)$$

Equation 4 shows the squares error where the error vector ($\sum_{ij} \boldsymbol{e_{ij}^{T}}(\boldsymbol{x_i}, \boldsymbol{x_j})$) relating the constraint i and j, and the information matrix error term $\Omega_{ij}\boldsymbol{e_{ij}}(\boldsymbol{x_i}, \boldsymbol{x_j})$ where we then can summarize into Equation 5. As shown in the above equation, only the nodes $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ contributes to the error term during that time and other nodes at that time will not be affected. Hence, least squares error is use as a solution for a system with many equations and unknowns whereby it minimizes the sum of the squared errors of the given system.

## 2. Methodology

In this section the methods that is used to approach and execute the research objective is further explained. It is divided into four sections whereby the first sections are explaining on how LIDAR data is being generated, follow by how did the generated LIDAR data is being used to carry out 2-D LIDAR slam, the improvement that is made on the pose graph optimization algorithm and lastly the evaluation of the obtained results.

### *2.1. LIDAR Data Generation*

A map of the environment was randomly drawn using Microsoft paint. The map was then uploaded into MATLAB and was built using the binary occupancy grid function, whereby the map of each cell represents the occupancy status of that cell. An occupied location is represented as true (1) and a free location is false (0). The parameters of the LIDAR sensor were set with a sensor range of 10 and a angel of - 3.1416 to + 3.1416. Besides, waypoints have also been plotted on the map as shown in Figure 3.

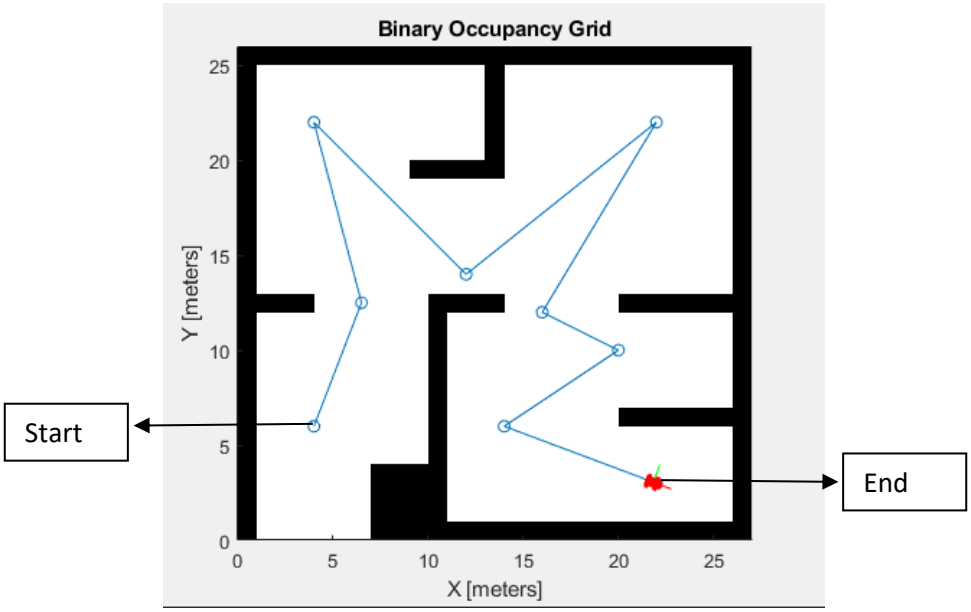



**Figure 3.** Waypoint generation to provide a path for the drone to navigate.

The waypoint that was plotted provides a path for the drone to navigate around the build map which acts a navigation simulator. As the drones navigate through the map according to the plotted

waypoints, it will start collecting LIDAR data for every single point. This LIDAR data is store in a cell 1 x 796 cell array and each cell contains the range data as well as the angles between each range. This shows that the drone has moved around the map with 796 points and at every point it stores the LIDAR data of the surrounding map.

*2.2. 2-D LIDAR SLAM via Pose Graph Optimization*

The LIDAR data that was obtained will then be used to carry our 2-D LIDAR SLAM by using the Pose Graph Optimization method. During pose graph optimization it creates a LIDAR-based slam object by incrementally taking in new LIDAR scans and attaches it to a node in the underlying pose graph. The slam algorithm also searches for new loop closures and reoptimize the node poses in the pose graph as new scans are added.

Firstly, the resolution of the occupancy grid map will be created based on the SLAM object and it is specified by the map resolution of number of cells per meter. This map act as a LIDAR-SLAM object and will be used as a platform to plot the poses as well as map the surrounding of the map simultaneously.

After creating a platform for the poses and LIDAR scans to be plotted. The LIDAR data will incrementally be added into the platform by undergoing pose graph optimization and plot the pose and map the surrounding simultaneously based on the LIDAR data that was generated. Since the LIDAR data generated has 796 poses it means that it has 796 nodes, and each node contains the LIDAR data and will go through the pose graph optimization algorithm for a total of 796 iterations to carry out the SLAM application. Figure 4 shows the pose and map that was plotted for 10 iterations which means 10 poses/nodes to provide an insight of how SLAM occurs.

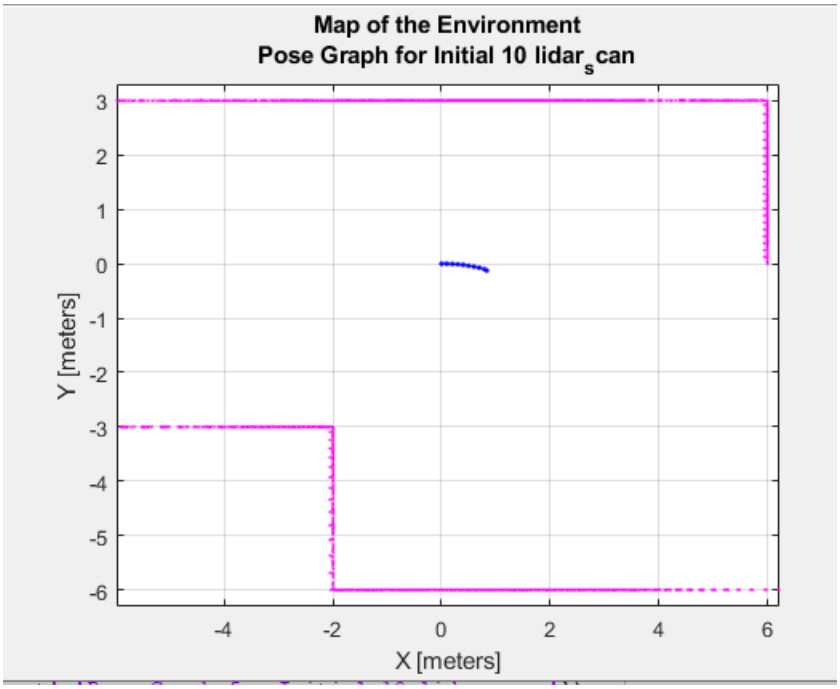



**Figure 4.** Pose graph for 10 initial LIDAR scans.

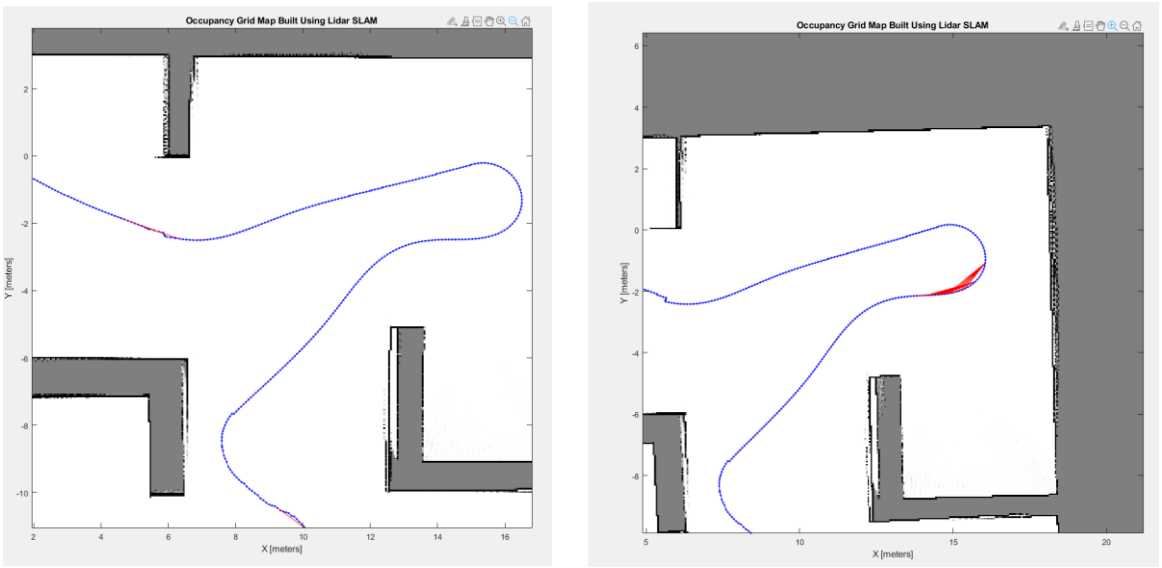
Hence, the steps of generating the poses and map the surrounding are as follows.

1) Creating a platform
2) Add scan data into the platform via pose graph optimization.
3) Remove loop closure.

4) Return scans and poses for given nodes.
5) Show overlaid scans and robot trajectory.

*2.3. Performance Evaluation*

The pose graph optimization codes were than further analyse and a loop closure threshold and loop closure search radius were found to affect the accuracy of the trajectory of the drone and mapping via pose graph optimization as well as the processing speed of obtaining the data. Firstly, the minimum and maximum values of the loop closure threshold and radius was determined. Which is 1 to 210 and 1 to 15, respectively. The reason of using the minimum and maximum value is to find out what are the changes in characteristics to the system when the value is at the most minimum and the most maximum. Firstly, the loop closure threshold, were set to a low value which is 1 and the loop closure search radius were also set to a value of 1. The time taken to run the code was recorded and the graph were being analysed. The steps were repeated with loop closure threshold of 210, and loop closure search radius of 15. It was found out that by increasing the loop closure threshold we are increasing the threshold of accepting loop closures during scan matching and increasing the number of loop closure constraints.




(a) (b)

**Figure 5.** (a) loop closure threshold of 1. (b) loop closure threshold of 210.

According to Figure 5 with higher loop closure threshold there will be more loop closure constraints that is shown by the dense red line which will then affect the amount of error that is going to be introduce by each constraint as each constraint contribute to 1 error. Hence, by decreasing the loop closure threshold we can reduce the loop closure constraints form which then helps to reduce the error during scan matching, which means it helps the nodes to easily agree with each other and the expected node to appear will be much more like the data constructed from odometry. The loop closure search radius on the other hand is known as the radius which the loop closure is search at every single node. If the search radius is large more nodes will be compared which then will increase the time taken of processing scan matching and will affect the overall SLAM.

*2.4. Analysis*

The trajectory of the drone via pose graph optimization and ground truth were plotted for both the y-plane and x-plane. The y-plane and x-plane represents the y-axis, and the number of data represents the x-axis. Hence, the trajectory of the drone via pose graph optimisation at a threshold of 1 and 210

were compared with ground truth. Besides the root mean square error (RMSE) of the trajectory were also calculated for both the x-axis plane and y-axis plane when comparing with ground truth. For each condition, the processing speed of carrying out SLAM via pose graph optimization method were run 5 times to obtain the average time taken for the SLAM to finish its data processing. Besides the percentage of the processing speed that were improved were calculated. The mapped surrounding that was done via pose graph optimization were obtained for both with loop closure threshold of 1 and loop closure threshold of 210. The map obtained were also compare with ground truth.

## 3. Results and Discussion

### *3.1. Processing Rate on Pose Graph Optimization*

**Table 1.** Average Time taken for 2-D LIDAR SLAM to be computed.

| Loop Closure Threshold | Loop Closure search radius (m) | Average time taken (s) | Processor | Number of iterations |
|---|---|---|---|---|
| 1 | 1 | 112.87 | Intel-core i7 | 796 |
| 1 | 15 | 203.88 | Intel-core i7 | 796 |
| 210 | 1 | 110.39 | Intel-core i7 | 796 |
| 210 | 15 | 201.97 | Intel-core i7 | 796 |

Based on Table 1, Loop closure threshold is the threshold for accepting a loop closure. This threshold is applied into the scan matching algorithm which helps to either improve the matching and vice versa. On the other hand, loop closure radius is the radius of the loop closure that is needed to be done in the given map. These two are independent of each other. By changing the loop closure search radius, we can change the time taken for the code to complete the mapping procedure of the surrounding environment and obtain the pose of the surrounding. Comparing the loop closure with search radius of 1 m and 15 m respectively with a loop closure threshold of 1 the processing time of the pose graph optimization is reduced from 203.88 s to 112.87 s. Hence, the processing rate can be decrease by 45.43%. Comparing the loop closure with search radius of 1 m and 15 m respectively with loop closure threshold of 210 the processing time of the pose graph optimization is reduced from 201.97 s to 105.39 s. Which shows that the processing rate can be decrease by 45.43%. Based on the percentage change in the processing rate which obtain from Equations (5) and (6) we can see that although at different loop closure threshold the percentage change in the processing rate is quite similar which shows that the loop closure threshold does not affect the processing rate.

### *3.2. Trajectory of Pose Graph Optimization*

**Table 2.** Root Mean Square Error when comparing with ground truth.

| Axis | Loop closure threshold | RMSE (m) |
|---|---|---|
| x | 1 | 2.10 |
| y | 1 | 10.97 |
| x | 210 | 3.97 |
| y | 210 | 13.75 |

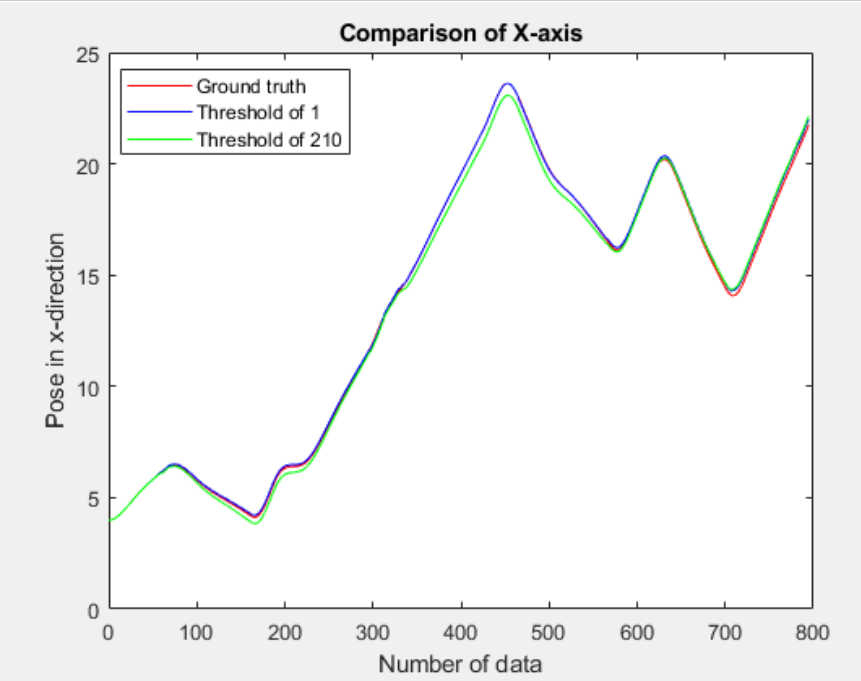



**Figure 6.** Trajectory of the robot at the x- axis of ground truth, threshold of 1 and threshold of 210.

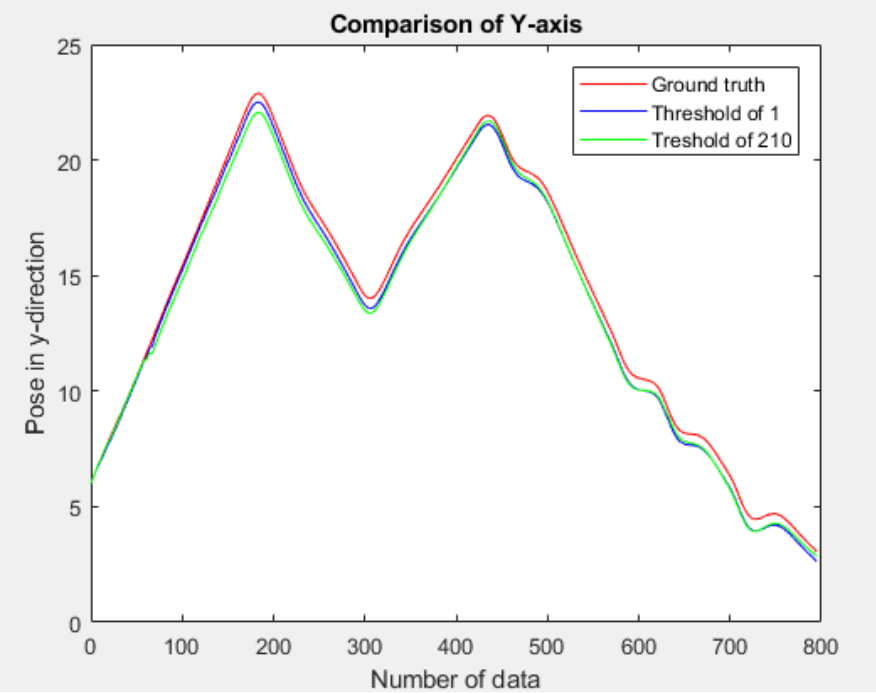



**Figure 7.** Trajectory of the robot at the y-axis of ground truth, threshold of 1 and threshold of 20.

Figure 6 and 7 shows comparison between the trajectory of the robot during ground truth, threshold of 210 and threshold of 1. By comparing the trajectory of the drone in the x-axis and y-axis with a threshold of 1 and 210 respectively, the results show that at a threshold of 1, the trajectory of the drone is much more accurate compared to the trajectory of the drone at a threshold of 210 at both the x-axis and y-axis. Besides, Figure 6 and Figure 7, also compares the trajectory of the drone at loop closure threshold of 210 and 1. It shows that at different threshold level, the trajectory of the drone is also different. Based on the results obtain as shown in table 2, at a threshold of 1 the x-axis and y-axis for each pose has a difference of 2.10 m and 3.97 m respectively between the calculated data when compared to ground truth. On the other hand, at a threshold of 210 the x-axis and y-axis for each pose has a difference of 10.97 m and 13 .75 m respectively between the calculated data when compared to ground truth. Therefore, the loop closure with lower threshold is more accurate with the loop closure with higher threshold based on the RMSE as shown in Table 2.

*3.3 Mapping*

This section shows and discusses on the mapped environment by using LIDAR data via pose graph optimization.

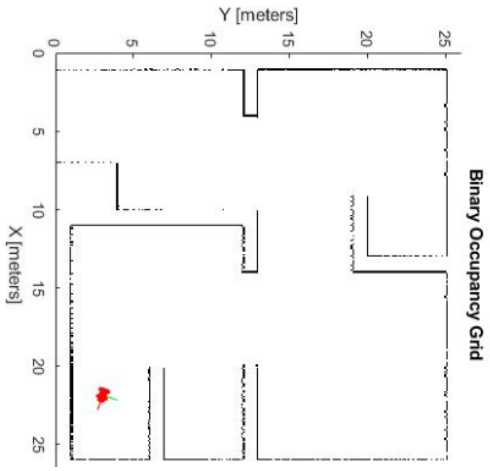



**Figure 8.** Ground truth Map

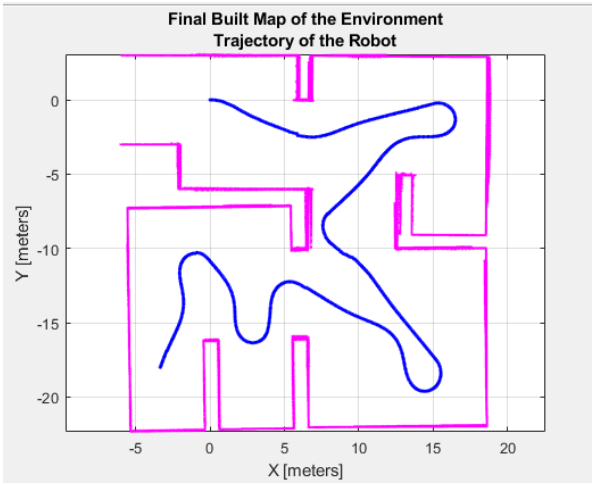



**Figure 9.** Map from pose graph optimization with threshold of 1

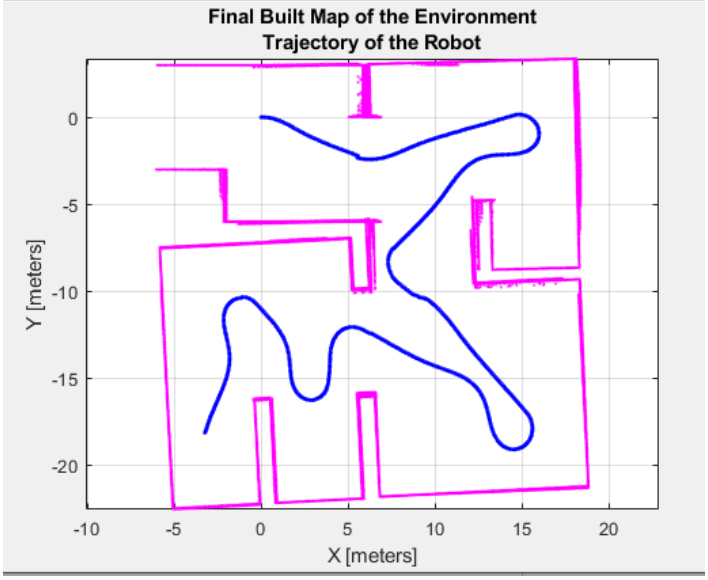



**Figure 10.** Map of pose graph optimization with threshold of 210.

Figure 8 shows the map of the surrounding that was generated by the LIDAR sensor and Figure 9 and 10 shows the map of the surrounding that was generated by the pose graph optimization with a threshold of 1 and 210, respectively. According to Figure 8 9 and 10 the results show that at a threshold of 1, the map of the environment generated through pose graph optimization is like ground truth compared to the map of the surrounding at a threshold of 210. This means that, the data obtain through pose graph optimization is closer to the original mapped surrounding as well as trajectory of the robot. Hence, it will not affect the performance of drone during flight time in real life application.

## 4. Conclusion

In this research the LIDAR data were able to be generated and were used to carry out LIDAR SLAM. The methods that were used to determine the pose and map the surrounding simultaneously is known as pose graph optimization. Pose graph optimization is a type of LIDAR SLAM where it is being presented in this research and it can return the pose of the drone and map the surrounding of the environment simultaneously by using the LIDAR data that was generated. In this pose graph optimization, the value of the loop closure threshold and loop closure search radius were being manipulated and the results were being observed. Based on the results obtained, it was found out that the loop closure threshold can manipulate the accuracy of the map as well as the trajectory of the drone by utilizing the LIDAR sensor data. On the other hand, the loop closure search radius can change the processing speed of the generated LIDAR data. By testing out these two variables, it is found that the loop closure threshold and loop closure radius are independent of each other. Hence, we can control the processing speed of carrying out SLAM as well as the accuracy of the data compared to ground truth. Therefore, pose graph optimization is consider as a robust method in the drone industry because it can control the processing speed rate of obtaining the data without affecting the accuracy of the data. It is very helpful because drone's data measurement is based on real-time. Hence, with any instance delay of accepting the data it will the affect the performance of the drone. However, this paper is limited as it is only using 2-Dimensional. Therefore, a 3-Dimensional SLAM could be done for other application and different scenarios.

## References

[1] Phang S K and Chen X 2021 Autonomous tracking and landing on moving ground vehicle with multi-rotor uav *J. Eng. Sci. Technol.* **16** 2795–815
[2] Herrero D, Villagra J and Martinez H 2013 Self-configuration of waypoints for docking maneuvers of flexible automated guided vehicles *IEEE Trans. Autom. Sci. Eng.* **10** 470–5
[3] Balamurugan G, Valarmathi J and Naidu V P S 2017 Survey on UAV navigation in GPS denied environments *Int. Conf. Signal Process. Commun. Power Embed. Syst. SCOPES 2016 - Proc.* 198–204
[4] Shamsuddin P N F M, Mansor M A, Ramli R M and Jaapar R M Q R 2020 Development of navigation system for unmanned surface vehicle by improving path tracking performance *J. Eng. Sci. Technol.* **15** 1371–83
[5] Tai J J, Phang S K and Wong Y M F 2020 Optimized autonomous UAV design with obstacle avoidance capability *AIP Conf. Proc.* **2233** 020026
[6] Slowak P and Kaniewski P 2020 LIDAR-based SLAM implementation using Kalman filter 5
[7] Ravankar A, Ravankar A, Kobayashi Y and Emaru T 2018 Autonomous Mapping and Exploration with Unmanned Aerial Vehicles Using Low Cost Sensors *Proceedings* **4** 44
[8] Yagfarov R, Ivanou M and Afanasyev I 2018 Map Comparison of Lidar-based 2D SLAM Algorithms Using Precise Ground Truth *2018 15th Int. Conf. Control. Autom. Robot. Vision, ICARCV 2018* 1979–83
[9] Ng Z Y and Phang S K 2020 Development of simultaneous localization and mapping algorithm using optical sensor for multi-rotor UAV *AIP Conf. Proc.* **2233** 030007
[10] Wang F, Cui J, Phang S K, Chen B M and Lee T H 2013 A mono-camera and scanning laser range finder based UAV indoor navigation system *2013 Int. Conf. Unmanned Aircr. Syst. ICUAS 2013 - Conf. Proc.* 694–701
[11] Khairudin M, Herlambang S P, Karim H I and Azman M N A 2020 Vision-based mobile robot navigation for suspicious object monitoring in unknown environments *J. Eng. Sci. Technol.* **15** 152–66
[12] Wang F, Wang K, Lai S, Phang S K, Chen B M and Lee T H 2014 An efficient UAV navigation solution for confined but partially known indoor environments *IEEE Int. Conf. Control Autom. ICCA* 1351–6