Lidar Based SLAM Algoritm

1st Ruihua Cai
Computer Engineering Department
San Jose State University
San Jose, CA, US
ruihua.cai@sjsu.edu

2nd Ziyuan Yan
Computer Engineering Department
San Jose State University
San Jose, CA, US
ruihua.cai@sjsu.edu

3rd Ruixiang Huang
Computer Engineering Department
San Jose State University
San Jose, CA, US
ruihua.cai@sjsu.edu

***Abstract*—In the field of mobile robotics, localization is a very important issue. One particularly difficult problem is how to enable a robot to determine its position in an unknown environment. Simultaneous localization and mapping (SLAM) is one way to solve this problem. Also, Lidar is the most common sensor used by mobile robots to implement SLAM. This project is intended to explore the difficulties of SLAM algorithms by implementing some simple 2D Lidar-based SLAM algorithms. This project will be simulated in Matlab to determine the accuracy and efficiency of the algorithm.**

***Index Terms*—Lidar, SLAM, ICP**

I. Introduction

In real-world robot research, Simultaneous localization and mapping (SLAM) is a critical topic for helping robots to understand their location in a space and create a map for collecting and analyzing the surrounding environment.

There are serval ways to implement SLAM on robots. Such as camera, sonar, and Lidar. Using the camera, we could use the image processing algorithm to get a graph of the current environment. Even though we could get a high-resolution image from it, due to the sensor itself doesn't provide very accurate distance data, and noise under too bright or dark conditions could affect the result. Camera-based SLAM is not common for people to research. Sonar-based SLAM is the majority applied to underwater robot SLAM. Due to the environmental issue, it could not rely on light-based sensors to perform SLAM. The best choice for underwater robots to do SLAM is based on sonar, which can handle the undercurrent in water [4]. Using Lidar is the most popular solution for solving SLAM issues. It could provide a wide range of searching angle and distance information, which is excellent for developing SLAM research.

Since Lidar is the most common and affordable device for SLAM research. By collecting the surrounding Lidar information, we could get the outline of the surrounding area. Due to the advantages of Lidar, which include the distance information, we could determine the distance between the robot and the wall or obstacles around the robot. In this project, we will develop two kinds of SLAM algorithms without the data of odometry. The first algorithm is based on the prediction robot's movement to match the similarity between the previous scan and the current scan result. If the robot could match the pattern of the scan result, we can say the robot moved in this direction and distance. The second algorithm is based on the Iterative Closest Point (ICP) algorithm. By comparing the closest point shifting between the previous scan and the current scan, we could get a vector shifting that is suitable for most of the points, and this vector is the movement of the robot.

II. Related Work and Background Material

By studying Wolfgang & Damon [1] and John & John [4]'s article, we have a more comprehensive understanding of the different sensor-based robot SLAM algorithms and how it benefits human life. This involved our interest in researching different kinds of SLAM approaches. In Shan's article [2], he demonstrates a way to implement SLAM by combining multiple sensor data and using the Extended Karmal Filter (EKF) to perform sensor fusion for calculating SLAM. We study how this author uses 2D-Lidar to collect the surrounding environment data and understand the basic theory of localization. Finally, from Shaofeng & Jingyu's article [3], we found another approach to implementing robot SLAM based on Lidar. Through the above research and study, these articles gave us basic knowledge and direction about how to implement SLAM into our project.

III. Technical approach

A. Pattern Match SLAM

The pattern match algorithm is our own design of the SLAM algorithm. We want to find the current position of the robot by assuming the displacement amount. Whenever a new scan is obtained, the program first speaks the result of the scan through a coordinate transformation to obtain a matrix with a set resolution. Each point in the matrix that has a lidar mapped to it is set to 1, and points that are not mapped are set to 0. We then convert the current scan by traversing the possible displacements and comparing the difference with the previous result. Then we obtain the difference between the two matrices by summing the absolute values of the difference matrix. By recording the displacement of the matrix with the smallest difference, we obtain the displacement between the robot this time and the last received result. Then by adding the previous coordinates to the current displacement, we obtain the correct coordinates. Since the first step of the algorithm is to obtain a matrix that can be calculated quickly by reducing the resolution of the scan. The accuracy of the algorithm gets better as the resolution increases, but because all possible displacements are calculated each time, the efficiency of the algorithm becomes slower as the resolution increases. To solve this problem, we add the parameter search distance to limit the number of displacements that the algorithm traverses. However, this also limits the distance of a single displacement of the robot, which can cause the algorithm to lose track if the robot is displaced too far.

B. Iterative Closest Point SLAM

The IEEEtran class file is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

IV. Results Analyze

Before you begin to format your paper, first write and save the content as a separate text file. Complete all content and organizational editing before formatting. Please note sections below for more information on proofreading, spelling and grammar.

Keep your text and graphic files separate until after the text has been formatted and styled. Do not number text heads—LaTeX will do that for you.

A. Abbreviations and Acronyms

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as IEEE, SI, MKS, CGS, ac, dc, and rms do not have to be defined. Do not use abbreviations in the title or heads unless they are unavoidable.

B. Equations

Number equations consecutively. To make your equations more compact, you may use the solidus (/), the exp function, or appropriate exponents. Italicize Roman symbols for quantities and variables, but not Greek symbols. Use a long dash rather than a hyphen for a minus sign. Punctuate equations with commas or periods when they are part of a sentence, as in:

$$a + b = \gamma \tag{1}$$

Be sure that the symbols in your equation have been defined before or immediately following the equation. Use “(1)”, not “Eq. (1)” or “equation (1)”, except at the beginning of a sentence: “Equation (1) is . . .”

C. Figures and Tables

a) Positioning Figures and Tables: Place figures and tables at the top and bottom of columns. Avoid placing them in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures; table heads should appear above the tables. Insert figures and tables after they are cited in the text. Use the abbreviation “Fig. 1”, even at the beginning of a sentence.

TABLE I
TABLE TYPE STYLES

Table Head	**Table Column Head**		
	Table column subhead	***Subhead***	***Subhead***
copy	More table copy[a]		

[a]Sample of a Table footnote.

Fig. 1. Example of a figure caption.

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity “Magnetization”, or “Magnetization, M”, not just “M”. If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write “Magnetization (A/m)” or “Magnetization $\{$A[m(1)]$\}$”, not just “A/m”. Do not label axes with a ratio of quantities and units. For example, write “Temperature (K)”, not “Temperature/K”.

V. Conclusion

Before you begin to format your paper, first write and save the content as a separate text file. Complete all content and organizational editing before formatting. Please note sections below for more information on proofreading, spelling and grammar.

Keep your text and graphic files separate until after the text has been formatted and styled. Do not number text heads—LaTeX will do that for you.

References

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

[2] S. Huang, H.-Z. Huang, Q. Zeng, and P. Huang, ”A Robust 2D Lidar SLAM Method in Complex Environment”, Photonic Sensors, vol. 12, no. 4, p. 220416, Apr. 2022.

[3] S. Wu and J. Lin, ”A novel SLAM framework based on 2D LIDAR”, Journal of Physics: Conference Series, vol. 1650, no. 2, p. 22104, 2020.

[4] J. Folkesson and J. Leonard, ”Autonomy through SLAM for an underwater robot”, in Robotics Research, Springer, 2011, pp. 55-70.