Low cost mobile navigation using 2D-SLAM in complex environments

Preprint · July 2023

DOI: 10.13140/RG.2.2.31842.50887

4 authors, including:

Ines Haouala
Università degli Studi di Genova
3 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Low cost mobile navigation using 2D-SLAM in complex environments

Aicha Manar ABBAD, Ines HAOUALA, Amanzhol RAISOV, Roumaissa BENKREDDA

***Abstract*—This research paper presents a low-cost mobile navigation system utilizing Simultaneous Localization and Mapping (SLAM) algorithms in complex dynamic environments. The study investigates the performance and effectiveness of four different SLAM algorithms, namely ORB-SLAM2, g-mapping, Hector SLAM, and karto SLAM algorithm. The objective is to develop a cost-effective solution that enables accurate and real-time mobile navigation in challenging scenarios. For that, the strengths and limitations of these four algorithms were discussed and compared to select the most appropriate algorithm that is suitable for the case of warehouse environment. Each algorithm is evaluated based on some metrics such as Map and Pose Accuracy, CPU and Memory Utilization, Robustness, and processing time to ensure their feasibility on low-cost mobile hardware.**

I. Introduction

SImultaneous Localization and mapping abbreviated as SLAM is one of the most fundamental issues in robotics. Researchers Peter Cheeseman, Jim Crowley, and Hugh Durrant-Whyte first presented it in 1986 at the IEEE Robotics and Automation Conference in San Francisco, where they were attempting to apply certain theoretical estimating approaches to mapping and localization [1]. The question posed by SLAM is whether it is possible to deploy a mobile robot in an uncharted environment while also creating a model map and pinpointing the robot's location inside it using various sensors. One significant issue with SLAM is that measurements obtained from the sensors always contain noise, and robot mobility also results in positional problems [2].

A. Purpose of this research paper

With the increasing needs of the global population, there is an eminent inclination towards Industry 4.0 and the use of new technologies. Latest research and investments were put to enhance the fields of agriculture, heavy industry, supply chain, etc. In the same context, autonomous systems gained large fame. Autonomous systems have proven to be efficient in handling repetitive and tedious tasks. With the emergence of Artificial Intelligence, they are better equipped to understand and interact with their environments. However, they have not yet reached the "stage of diffusion" as stated in [3]. One of the prominent challenges is the complexity of the workspace. There is a rising demand for robots to become smarter and better understand and adapt to their environments. This problem has received substantial interest and remains an open problem in the area. This has been previously assessed only to a relatively limited extent because previous research was constrained to the development of new SLAM algorithms with no major focus on dynamic and complex environments and no specific example was taken. For the sake of this research, the example of the warehouse is taken as an example of a complex environment where a mobile has to successfully navigate and complete its tasks. Warehouses are constantly congested with operators, packages, and lift trucks. They have high flow activity and therefore present numerous obstacles to the robots. In some cases, autonomous robots fail to correctly interpret information from their sensors and act adequately. The most dreadful situation is when the robot is unable to understand changes in its physical world and falls into a deadlock state, and freezes. While this is easily solved with human intelligence, this is a very common problem in robotics research. Consequently, autonomous systems do not operate without constraints and are not yet fully feasible. The aim of this research paper is to look for the best solution for mobile robots to successfully navigate in a complex environment with a focus on warehouses. It will analyze and compare four different simultaneous localization and mapping algorithms to ensure safe navigation of ground robots supervising a warehouse. Ultimately, a deadlock state should be prevented in crowded spaces. For this application, the robots should have the ability to cooperate and coordinate in terms of manipulation and perception in real-time to complete tasks such as monitoring, pick-and-place goods, identifying technical anomalies, etc. In more complex tasks, they will be required to cooperate with humans with high accuracy and safety. To do this, the paper will explore four of the most widely used 2D SLAM solutions in indoor environments. These concepts will help the robots make more informed decisions according to the situation and maintain constant communication.

II. Methodology

A. Explanation of the selection criteria for the SLAM approach

As mentioned in the previous section, the focus of this research will be on a complex environment and to be more specific on the warehouse. It is important to note that warehouses are constantly congested and busy environments. Examples of large warehouses are the Tesla Gigafactory 1 in the United States with 492k m2 or Amazon Fulfilment Centre with 492k m2 [4]. They are also considered dynamic since there are agents and other vehicles constantly moving and transporting goods. In the context of mobile navigation, warehouses present many obstacles such as corners or sharp objects, different lightening conditions, in-motion, unpredictable obstacles, etc. Our research aims at finding a solution to this challenging problem. This paper examines some previous work and investigates the efficiency of four different SLAM algorithms. The main objective is to accomplish a comparison analysis and deduct the best method for our use case. To do so, the following five metrics were selected to evaluate the 2DSLAM algorithms and choose the optimal one for the use in warehouse:

- Map Accuracy: This metric measures how accurately the algorithm maps the area. This is an effective technique to assess through visual inspection the consistency, entireness and correctness of the map. It can be evaluated by comparing the generated diagram with the ground truth diagram [5].
- Pose Accuracy or relative pose error (RPE): Based on the result pose of the robot, this metric measures the algorithm's accuracy in estimating the correct pose. The RPE is retrieved from the motion error between pairs of timestamps in the estimated trajectory file with the ground truth [6].
- CPU and Memory Utilization: This is to measure the use of CPU and memory usage by the selected algorithm. Overall, it measures the computational resources necessary by the algorithm to run SLAM [5].
- Robustness: This is to test and demonstrate the performance and consistency of the algorithm in challenging environments such as complex and dynamic environments. This metric can be used to test the algorithms in different environments and asses their behavior and performance [5].
- Processing Time: This is also another important metric to consider while evaluating SLAM algorithms. This is based on measuring the processing time and real-time performance of the SLAM [7].

III. Review of the SLAM Algorithms

A. Gmapping Algorithm

1) Overview: The Gmapping algorithm is the one of the most used algorithms for 2D-SLAM using laser radar for indoor navigation. The Gmapping algorithm achieves the process of position estimation and map creation by introducing the RBpf particle filtering algorithm, which effectively separates the tasks of positioning and mapping. First, the RBpf particle filtering algorithm is used to obtain the optimal pose of the robot and localize it. Then, based on the Kalman filter, the map is constructed. The formula for this process is [8]:

$$\mathrm{p}\left(x_{1:t},\ \mathrm{m} \mid z_{1:t}, u_{1:t-1}\right) = \mathrm{p}\left(\mathrm{m} \mid x_{1:t}, z_{1:t}\right) * \mathrm{p}\left(x_{1:t} \mid z_{1:t}, u_{1:t-1}\right) \tag{1}$$

The robot's posture algorithm consists of four steps that calculate its position continuously: resampling, weight calculation, map estimation, and sampling. The formula x represents the trajectory of the robot, while the observation data from the lidar is denoted as z. The characteristic map obtained from the lidar observation is represented as m, and the odometer measurement value is represented as u. Iteratively, this algorithm determines the robot's posture [8].

2) Advantages of Gmapping: Gmapping applies the RBpf algorithm to optimize the suggested distribution, resulting in a transformation of the formula. In order to achieve this, a threshold is established to selectively choose particles requiring resampling, leading to a reduction in the number of particles, as show in the formula below. As a consequence, the gmapping algorithm requires less computational and memory use [8].

$$\mathrm{p}\left(\mathrm{x_t} \mid \mathrm{m}_{\mathrm{t}-1}^{(\mathrm{i})}, \mathrm{x}_{\mathrm{t}-1}^{(\mathrm{i})}, \mathrm{z_t}, \mathrm{u}_{\mathrm{t}-1}\right) = \frac{\mathrm{p}\left(\mathrm{z_t} \mid \mathrm{m}_{\mathrm{t}-1}^{(\mathrm{i})}, \mathrm{x_t}\right) \mathrm{p}\left(\mathrm{x_t} \mid \mathrm{x}_{\mathrm{t}-1}^{(\mathrm{i})}, \mathrm{u}_{\mathrm{t}-1}\right)}{\mathrm{p}\left(\mathrm{z_t} \mid \mathrm{m}_{\mathrm{t}-1}^{(\mathrm{i})}, \mathrm{x}_{\mathrm{t}-1}^{(\mathrm{i})}, \mathrm{u}_{\mathrm{t}-1}\right)} \tag{2}$$

The utilization of Gmapping presents a reliable resource for mobile robots, providing them with the ability to attain precise mapping and localization capabilities. This advanced tool employs a grid-based approach to effectively depict new objects, ultimately leading to the creation of precise maps. Li Shao et al. [8] have tested Gmapping algorithm in a flat indoor environment with the following parameters, speed of robot was 0.5 m/s, angular velocity, 1 rad/s, and map resolution of 0.05. The simulation was run on Gazebo ROS tool. Li Shao et al. demonstrated that Gmapping provides an accurate and complete map, similar to the real simulation environment [8].

The Gmapping algorithm demonstrates a notable capacity in dealing with uncertainties. It accomplishes this by utilizing probabilistic techniques to tackle uncertainties that emerge from both the movement of the robot and the measurements of its sensors. As mentioned earlier, Gmapping employs a strategy based on particle filters, allowing it to proficiently navigate through non-linearities and uncertainties linked to the robot's motion and sensor models. Bo Li et al. [9] demonstrated in their experiment that Gmapping gives high-quality details where corners were well reproduced in the map despite the complexity of the environment.

Moreover, Gmapping offers the possibility to fine-tune its parameters to suit specific requirements. This flexibility enables optimal performance across different environments and with a wide range of robot platforms. By adjusting variables such as the number of particles, sensor noise, and map resolution, developers have the ability to customize the algorithm and attain their desired level of performance [10].

3) Limitations and challenges of Gmapping: The computational demands of Gmapping become important when dealing with scenarios that involve a multitude of particles or intricate maps. The algorithm necessitates substantial computational power to handle sensor data, continuously update the map, and accurately estimate the robot's position in real-time. In [9], it was shown that in complex environments the Gmapping algorithm should be improved by increasing the number of particles which leads to heavy computational cost.

Moreover, Gmapping relies on accurate sensor measurements for mapping and localization. It can be sensitive to sensor noise, especially in situations where the sensor data is noisy or inconsistent. Noisy sensor data can lead to inaccuracies in the generated map and the estimated robot pose. Nwankwo Linus et al. [5] showed that Gmapping heavily relies on the perfect accuracy of the odometry sensor to report back on the robot's pose. However, the odometry is prone to errors, particularly in complex environments and due to other factors such as wheel slippage, sensor noise, etc [5].

It was also proven that Gmapping assumes a static and an indoor environment during the mapping process. It does not handle dynamic objects or outdoor environments well. Moving objects or changes in the environment can introduce errors in the map and affect the accuracy of localization [11].

B. ORB-SLAM2 Algorithm

1) Overview: Is an advanced simultaneous localization and mapping (SLAM) algorithm that provides accurate and real-time camera trajectory and sparse 3D reconstruction in various environments due to its effectiveness in diverse scenarios ranging from small-scale indoor scenes to large-scale outdoor scenarios , it is designed to work with different types of cameras including monocular, stereo, and RGB-D cameras which made it able to handle various sensor configurations , it is also able to handle loop closures ,perform global relocalization ,and to automatically in a robust way initialize from both planar (2D) and non-planar (3D) scenes [12] . According to [12], [13] , To execute simultaneous localization and mapping (SLAM) utilizing visual data, ORB-SLAM2 is made up of several key components that function together. Here is a list of its essential elements:

- feature detection and matching : The ORB (Oriented FAST and Rotated BRIEF) feature detector and descriptor are used by ORB-SLAM2 for feature detection and matching. It uses the BRIEF (Binary Robust Independent Elementary Features) algorithm to compute a binary descriptor and the FAST (Features from Accelerated Segment Test) approach to identify keypoints in the image. These ORB characteristics are excellent for reliable tracking since they are scale- and rotation-invariant.
- Tracking: The component that tracks calculates the camera's pose (position and orientation) in relation to the starting frame in each frame. It implements the RANSAC (Random Sample Consensus) algorithm to estimate the camera motion and searches for matches between subsequent frames using the detected ORB features. In order to improve the posture estimation, a local bundle adjustment is used.
- Local Mapping: Using chosen keyframes, the local mapping component creates a local map. Keyframes are selected based on how they move in relation to the keyframe before them. The tracked features' 3D positions are estimated for each keyframe using the ORB features and their descriptors. In order to reduce errors, the relative camera postures between keyframes are simultaneously optimized via bundle adjustment.
- Loop Closing: Loop Closing in ORB-SLAM2 identifies when the camera revisits a place, comparing the current frame with earlier keyframes for consistent matches. It uses global bundle adjustment to correct the map and improve accuracy by fixing accumulated errors.
- Map Representation: In 2D SLAM, the map is represented by ORB-SLAM2 as a collection of keyframes and 2D points. Keyframes store the camera poses and feature descriptors, while the 2D points record the estimated 2D positions of the tracked features. The map is incrementally updated and optimized using bundle adjustment to improve accuracy and consistency.
- Relocalization: In the event of tracking failure or re-entering a previously mapped area, relocalization is the process of re-estimating the camera pose. Using a bag-of-words method, ORB-SLAM2 swiftly relocalizes and recovers tracking by comparing the current frame to a database of keyframes.
- Map Management and Optimization: In order to preserve efficiency, ORB-SLAM2 controls the map by removing extraneous keyframes and 2D points. In order to better optimize the entire map and improve the camera poses and 2D point placements, it also periodically executes global bundle adjustment.

Fig. 1 is an illustration of the actual algorithm [12].

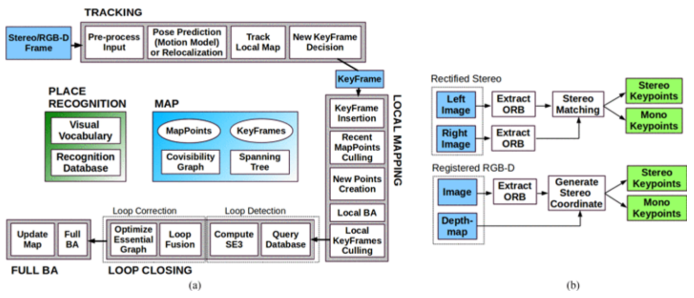

Fig. 1. ORB-SLAM2 algorithm

2) Advantages of ORB-SLAM2: As any other algorithm, ORB-SLAM2 has advantages that distinguish it. These are its main points of differentiation and strong points: [14]

- Strong Tracking: The usage of the ORB feature detector and descriptor allows ORB-SLAM2 to display strong tracking abilities. The design of ORB features makes them very resistant to changes in viewpoint and lighting since they are invariant to scale and rotation. [14]
- Real-Time Performance: The ORB-SLAM2 algorithm is renowned for its real-time performance, which enables it to offer 3D reconstruction and live camera tracking. Even on computers with limited resources, it uses effective methods and optimizations, such as multi-threading, to produce high frame rates. [14]
- Support for Monocular, Stereo, and RGB-D Setups: Due to its adaptability, it can handle a variety of sensor combinations and is compatible with a variety of hardware platforms and applications. [14]
- Map optimization and loop closure detection are features that ORB-SLAM2 incorporates to recognize revisited sites and fix accumulated inaccuracies. The algorithm optimizes the entire trajectory and map during loop closure, increasing the consistency and accuracy of the map by performing global bundle adjustment [14].
- Scalability and map management: ORB-SLAM2 efficiently controls the map while reducing computational complexity by selecting keyframes for optimization. This lowers the computational complexity while producing maps of good quality. Using map culling methods, it

controls the number of keyframes and 2D points, ensuring scalability and efficient memory use [14].

- Open-source and in active development: ORB-SLAM2 is a project that is supported by a lively research community. It has an extensive user base and benefited from ongoing community development, bug fixes, and feature updates [14].

3) Limitations and challenges of ORB-SLAM2:

- Vocabulary Needed: A trained bag-of-words vocabulary is used by ORB-SLAM2 for feature matching and loop closure detection. This suggests that developing an initial vocabulary is necessary, which calls for training on a sample dataset [14].
- Dependence of Localization Mode on Relocation: In ORB-SLAM2, the localization mode depends significantly on the capability of relocalization inside the map. The ability of the algorithm to maintain precise localization may be hampered if the camera loses track or fails to accurately relocalize [14].
- Unknown Absolute Scale in Monocular Mode: In ORB-SLAM2's monocular mode, the reconstructed map's absolute scale is unknowable. Due of their depth information, the stereo and RGB-D components may estimate scale, but the monocular component is unable to do so, which limits its use in applications requiring precise scale estimation [14].

C. KARTO-SLAM Algorithm

1) Overview: Karto SLAM algorithm is one of the algorithms that works based on the graph-based method to localise the vehicle or the AGV and to perform mapping [15]. The Karto SLAM utilises the current pose and the complete map along with the adding or updating the pose estimate. This makes the Karto SLAM an efficient SLAM than the others. The major disadvantage with the Kato SLAM is that the complexity of computation is proportional to the number of landmarks. As the number of landmarks increases, the computational complexity also increases

Karto SLAM [16] is a graph optimization method that uses a highly optimised and non-iterative cholesky matrix for sparse system decoupling as a solution. The mean of the graph is used to represent the map, and each node represents a location point of the robot trajectory and the data set of sensor measurement. The arrow pointing to the connection represents the movement of the continuous robot position point. When a new node joins, the map will be calculated and updated according to the constraint of the node arrow in the space. The ROS version of Karto SLAM, which uses the Spare Pose Adjustment (SPA), is related to scan matching and loop closure detection. The more landmarks, the greater the memory requirements. However, the graph optimization method has a greater advantage than other methods in a large environment. In some cases, Karto SLAM is more effective because it only contains a point graph (robot pose) [17]. The framework of Karto SLAM algorithm is shown in Figure 2.

2) Advantages of Karto SLAM: Generally, the term Karto SLAM is used to refer to a traditional laser based algorithm

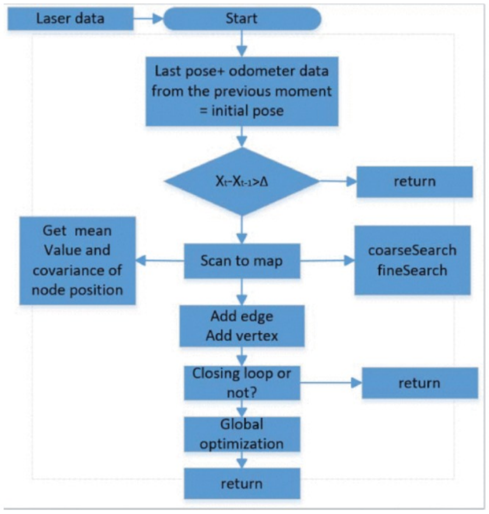

Fig. 2. Framework of karto SLAM algorithm

that has been widely adopted for various applications. In order to solve the problem of matrix direct solutions in nonlinear optimization, which is efficient with respect to convergence speeds and accuracy, Karto applies a sparse pose adjustment. In addition, Karto SLAM is able to map more quickly than others that have a lower error rate [18]. The specialization in 2D mapping, which enables the solution to be more effective and portable than SLAM algorithms that can map 3D objects, is one of the most important benefits of Karto SLAM. In this way, with a focus on 2D environments, Karto SLAM is able to operate at an optimal level for scenarios in which full 3D representation of the environment does not have to be required, leading to more resource friendly mapping processes [19]. The second benefit of Karto SLAM is to provide multiplatform support by implementing C++ and making it compatible with a variety of operating systems and embedded devices. This feature enables users to use this algorithm in a seamless manner across the range of platforms, so that it can be used for many different robotics and autonomously operated systems applications .

3) Disadvantages of Karto SLAM: As the first disadvantage, we can say that SLAM, first of all, focused on 2D mapping, which means that it is not well suited to applications that require 3D mapping. However, the level of detail necessary for some scenarios requiring a full 3D environment representation could not be supplied by it as a specific 2D SLAM algorithm [19]. Second, it won't be possible to take some benefit of its real-time performance at the time when the Karto algorithm gives exact results, in a wide environment. In such kind of cases, the processing speed could cause a potential problem for time-critical situations by requiring fast mapping and localization; this as a result will limit its suitability [18]. The third disadvantage is that Karto SLAM had relatively limited open-source development and maintenance compared to some of the other SLAM libraries. This could make it more difficult to update and possibly reduce the amount of support in the community for this algorithm [19].

D. Hector-SLAM Algorithm

1) Overview: Hector-SLAM is a mapping algorithm designed for laser range finder sensors like LIDAR (Light Detection and Ranging). This algorithm uses the grid mapping method for mapping an unknown environment, and pose matching to locate the robot within that environment [20]. The algorithm starts with some initial guess of the robot's position and starts to constantly refine the robot's pose by aligning the laser scan measurements with the provided map, this updates the occupancy grid and adjusts the estimated pose.

Grid mapping is the concept of dividing the environment around into a grid of cells where the occupancy of each cell is set to either free, occupied, or unknown.

Scan matching consists of successively aligning laser scan data to enhance the accuracy of the robot's estimated position.

The target is to maximize the probability of the grid occupied [21]:

In latex:

$$E(T) = \arg\min_{T} \sum [1 - M(S_i(x_T))]^2 \tag{1}$$

In latex:

$$S_i(T) = \begin{bmatrix} \cos T_\theta & -\sin T_\theta & T_x \\ \sin T_\theta & \cos T_\theta & T_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{ix} \\ p_{ij} \\ 1 \end{bmatrix} \tag{2}$$

T is the pose of millimeter wave radar.

S_i is the coordinate of laser point i after converting to T.

M is the probabilistic occupancy map.

P_i is the coordinate of laser point.

2) Advantages of Hector SLAM: - One of the remarkable benefits of Hector SLAM is its real-time functionality, which makes it exceptionally well-suited for various applications involving robots that need to map and navigate their environments in real-time. This capability is particularly advantageous in scenarios such as drone operations. where swift and accurate mapping and navigation are essential for efficient and successful tasks [22].

-Odometry is not required: Hector SLAM does not need wheel encoders or other odometry data to determine its location, which lessens the requirement for accurate motor control and lowers the possibility of cumulative errors [23].

3) Disadvantages of Hector SLAM: -The Hector approach, while not relying on odometer data, does exhibit some map drift. However, it compensates for this limitation by generating a highly detailed and comprehensive map of the environment [23]. -Hector SLAM may experience accumulated drift over time, which could result in inaccurate map generation and pose estimation [23]. This can be reduced by using loop-closure detection methods or applying other long-term drift-tolerant SLAM algorithms. -Sensitivity to sensor limitations: The quality and properties of the utilized sensor, such as its limited range, noisy data, or its susceptibility to changing ambient circumstances, might have an impact on performance [23].

IV. Results

1) Identification of the most efficient SLAM algorithm based on the evaluation: After knowing the strenghts and weaknesses of each algorithm , a comparison is done based on the four metrics mentioned bellow , to finally evaluate the best algorithm for our use case (warehouse):

Map Accuracy: ORB-SLAM2 is known for its accurate mapping capabilities, making it the best choice for map accuracy among the four algorithms.

Pose Accuracy or Relative Pose Error (RPE):For accurate pose estimation in 2D environments, the choice should be based on algorithms that are specifically designed and optimized for 2D mapping. g-mapping and Karto SLAM are both 2D SLAM algorithms, and they may be better suited for pose accuracy in 2D scenarios compared to ORB-SLAM2.

CPU and Memory Utilization: ORB-SLAM2 is again the best choice in this metric due to its efficient utilization of computational resources, making it suitable for low-cost mobile hardware.

Robustness: ORB-SLAM2 and g-mapping are both robust algorithms, but ORB-SLAM2's versatility in handling loop closures, relocalization, and initialization from both 2D and 3D scenes gives it an edge in this criterion.

Processing Time: ORB-SLAM2 excels in processing time with its real-time performance, making it the best choice for this metric as well.

Based on these metrics, ORB-SLAM2 tends to be the best-performing algorithm. However, it's essential to note that the best choice of algorithm ultimately depends on the specific requirements of the application, hardware constraints, and cost considerations. Further testing on real-world data are necessary to validate the performance of each algorithm in the specific environment(warehouse).

2) Limitations of the study and Suggestions for future research or improvements : The research paper presented a comprehensive evaluation of four different SLAM algorithms for mobile navigation in complex dynamic environments. However, the study has certain limitations that should be acknowledged: 1. Lack of Real-time Testing: One significant limitation is that the algorithms were not tested in real-time on a concrete environment or a practical platform like MATLAB or ROS. Real-world scenarios may introduce additional challenges and uncertainties that cannot be fully captured in simulation or offline testing. To overcome this limitation, future work should focus on implementing and testing the algorithms on actual mobile robots in dynamic environments. 2. Hardware Constraints: The research mainly focused on evaluating the computational requirements and memory usage of each algorithm to ensure feasibility on low-cost mobile hardware. However, the specific hardware used for testing was not clearly defined. To provide more practical insights, future studies should conduct experiments on a variety of low-cost mobile robot platforms with different hardware capabilities. 3. Specific Environment: The study used a warehouse environment as an example of a complex dynamic scenario. While this setting is relevant for applications like logistics and warehousing, the findings might not fully translate to other dynamic environments, such as outdoor settings or industrial facilities. To establish broader applicability, additional tests in various dynamic environments should be conducted.

V. Conclusion

In conclusion, this research paper focused on developing a low-cost mobile navigation system using Simultaneous Localization and Mapping (SLAM) algorithms for complex dynamic environments, with a particular emphasis on warehouses. The study evaluated four SLAM algorithms: ORB-SLAM2, g-mapping, Hector SLAM, and Karto SLAM, aiming to select the most appropriate algorithm for accurate and real-time mobile navigation without relying on expensive external sensors. The research considered several key metrics, including map accuracy, pose accuracy, CPU and memory utilisation, robustness, and processing time, to assess the performance of each algorithm. Based on the evaluation, ORB-SLAM2 emerged as the top-performing algorithm, showcasing strengths in map accuracy, CPU and memory utilisation, robustness, and processing time. It excels in real-time performance and handles loop closures and relocalization effectively. Despite ORB-SLAM2's strong performance, the selection of the best algorithm must consider specific application requirements, hardware limitations, and cost considerations. In some scenarios, g-mapping and Karto SLAM, being 2D SLAM algorithms, may be more suitable for pose accuracy in 2D environments. The research findings present valuable insights for developing cost-effective mobile robots that can navigate successfully in complex environments like warehouses. However, real-world testing and validation are necessary to confirm the algorithm's performance and effectiveness in the targeted application. Further development and community support for the chosen algorithm will be essential to enhance its functionalities and optimise its performance for real-world deployment. Overall, this research contributes to the advancement of autonomous systems, particularly in industrial settings, where intelligent mobile robots play a crucial role in enhancing efficiency and productivity.

References

[1] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. IEEE robotics automation magazine, 13(2):99–110, 2006

[2] Abu Bakar Sayuti HM Saman and Ahmed Hesham Lotfy. An implementation of slam with extended kalman filter. In 2016 6th International Conference on Intelligent and Advanced Systems (ICIAS), pages 1–4. IEEE, 2016.

[3] S. Yaghoubi, N. Ali Akbarzadeh, S. S. Bazargani, S. S. Bazargani, M. Bamizan, and M. I. Asl,"Autonomous Robots for Agricultural Tasks and Farm Assignment and Future Trends in Agro Robots," International Journal of Mechanical and Mechatronics Engineering IJMME-IJENS, vol. 13, no. 03, doi: 10.1.1.418.3615.

[4] N. Emmott, "Top 20 largest warehouses in the world," Avanta UK, https://www.avantauk.com/top-14 largest-warehouses-in-the-world/.

[5] L. Nwankwo and E. Rueckert, "Understanding why slam algorithms fail in modern indoor environments," Advances in Service and Industrial Robotics, pp. 186–194, 2023. doi:10.1007/978-3-031-32606-6-22

[6] D. Prokhorov, D. Zhukov, O. Barinova, K. Anton, and A. Vorontsova, "Measuring robustness of Visual slam," 2019 16th International Conference on Machine Vision Applications (MVA), 2019. doi:10.23919/mva.2019.8758020

[7] L. Fahle, E. A. Holley, G. Walton, A. J. Petruska, and J. F. Brune, "Analysis of SLAM-based Lidar Data Quality Metrics for geotechnical underground monitoring," Mining, Metallurgy amp; Exploration, vol. 39, no. 5, pp. 1939–1960, 2022. doi:10.1007/s42461-022-00664-3

[8] C. Wang, X. He, H. Ma, and L. Shao, "A comparative study of several mapping algorithms based on Laser Slam," Lecture Notes in Electrical Engineering, pp. 14–24, 2022. doi:10.1007/978-981-19-3927-3-2

[9] J. Ruan, Z. Fang, B. Li, Y. Wang and W. Zhao, "Evaluation of GP-SLAM in real-world environments," 2019 Chinese Automation Congress (CAC), Hangzhou, China, 2019, pp. 3076-3081, doi: 10.1109/CAC48633.2019.8996403.

[10] C. Tian, H. Liu, Z. Liu, H. Li and Y. Wang, "Research on Multi-Sensor Fusion SLAM Algorithm Based on Improved Gmapping," in IEEE Access, vol. 11, pp. 13690-13703, 2023, doi: 10.1109/ACCESS.2023.3243633.

[11] S. Liu, Y. Lei and X. Dong, "Evaluation and Comparison of Gmapping and Karto SLAM Systems," 2022 12th International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER), Baishan, China, 2022, pp. 295-300, doi: 10.1109/CYBER55403.2022.9907154.

[12] M. K. Rohil and A. Prakash, "Exploring Possible Applications of ORB SLAM 2 in Education, Healthcare, and Industry: Insights into the Challenges, Features, and Effects," 2023 IEEE 8th International Conference for Convergence in Technology (I2CT), Lonavla, India, 2023, pp. 1-7, doi: 10.1109/I2CT57861.2023.10126239.

[13] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," in IEEE Transactions on Robotics, vol. 33, no. 5, pp. 1255-1262, Oct. 2017, doi: 10.1109/TRO.2017.2705103.

[14] Acosta-Amaya, G.A., Cadavid-Jimenez, J.M. and Jimenez-Builes, J.A. (2023) 'Three-dimensional location and mapping analysis in Mobile Robotics based on Visual Slam Methods', Journal of Robotics, 2023, pp. 1–15. doi:10.1155/2023/6630038.

[15] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent, "Efficient sparse pose adjustment for 2D mapping," IEEE/RSJ 2010 Int. Conf. Intell. Robot. Syst. IROS 2010 - Conf. Proc., no. October, pp. 22–29, 2010

[16] Kurt Konoli, Giorgio Grisetti, Rainer Kummerle, Benson Limketkai and Regis Vincent, "Efficient Sparse Pose Adjustment for 2D mapping[J]", 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, December 2010.

[17] Z. Xuexi, L. Guokun, F. Genping, X. Dongliang and L. Shiliu, "SLAM Algorithm Analysis of Mobile Robot Based on Lidar," 2019 Chinese Control Conference (CCC), Guangzhou, China, 2019, pp. 4739-4745, doi: 10.23919/ChiCC.2019.8866200

[18] B. Zhang, M. Zhu, C. Lin and D. Zhu, "Research on AGV map building and positioning based on SLAM technology," 2022 IEEE 5th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE), Shenyang, China, 2022, pp. 707-713, doi: 10.1109/AUTEEE56487.2022.9994471.

[19] Trejos K;Rincón L;Bolaños M;Fallas J;Marín L; (no date) 2d Slam algorithms characterization, calibration, and comparison considering pose error, map accuracy as well as CPU and memory usage, Sensors (Basel, Switzerland). Available at: https://pubmed.ncbi.nlm.nih.gov/36146253/ (Accessed: 21 July 2023).

[20] S. Nagla, "2D Hector SLAM of Indoor Mobile Robot using 2D Lidar," 2020 International Conference on Power, Energy, Control and Transmission Systems (ICPECTS), Chennai, India, 2020, pp. 1-4, doi: 10.1109/ICPECTS49113.2020.9336995.

[21] R. Pan, Q. Sun, L. Xing, F. Peng and X. Wang, "Comparison of Scene Reconstruction Methods Based on Hector SLAM," 2022 3rd China International SAR Symposium (CISS), Shanghai, China, 2022, pp. 1-4, doi: 10.1109/CISS57580.2022.9971285.

[22] D. -T. Ngo and H. -A. Pham, "Towards a Framework for SLAM Performance Investigation on Mobile Robots," 2020 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea (South), 2020, pp. 110-115, doi: 10.1109/ICTC49870.2020.9289428.

[23] Z. -X. Li, G. -H. Cui, C. -L. Li and Z. -S. Zhang, "Comparative Study of Slam Algorithms for Mobile Robots in Complex Environment," 2021 6th International Conference on Control, Robotics and Cybernetics (CRC), Shanghai, China, 2021, pp. 74-79, doi: 10.1109/CRC52766.2021.9620122.