Comparison of Different Hardware Configurations for 2D SLAM Techniques Based on Google Cartographer Use Case

Łukasz Sobczak (✉ lukasz.sobczak@polsl.pl)

Silesian University of Technology

Katarzyna Filus

Polish Academy of Sciences

Joanna Domańska

Polish Academy of Sciences

Adam Domański

Silesian University of Technology

Research Article

Keywords: Robotics, SLAM, hardware configurations, framework

Posted Date: December 28th, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-1181853/v1>

License: This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Comparison of different hardware configurations for 2D SLAM techniques based on Google Cartographer use case

Łukasz Sobczak^{1,2,*}, Katarzyna Filus³, Joanna Domańska³, and Adam Domański²

¹OBRUM Sp. z o.o., Simulation Departement, Gliwice, 44-117, Poland

²Silesian University of Technology, Faculty of Automatic Control, Electronics and Computer Science, Department of Distributed Systems and Informatic Devices, Gliwice, 44-100, Poland

³Institute of Theoretical and Applied Informatics, Polish Academy of Sciences, Gliwice, 44-100, Poland

*lukasz.sobczak@polsl.pl

ABSTRACT

One of the most challenging topics in Robotics is Simultaneous Localization and Mapping (SLAM) in the indoor environments. Due to the fact that Global Navigation Satellite Systems cannot be successfully used in such environments, different data sources are used for this purpose, among others LiDARs (Light Detection and Ranging), which have advanced from numerous other technologies. Other embedded sensors can be used along with LiDARs to improve SLAM accuracy, e.g. the ones available in the Inertial Measurement Units and wheel odometry sensors. Evaluation of different SLAM algorithms and possible hardware configurations in real environments is time consuming and expensive. For that reason, in this paper we evaluate the performance of different hardware configuration used with Google Cartographer SLAM algorithms in simulation framework proposed in¹. Our use case is an actual robot used for room decontamination. The results show that for our robot the best hardware configuration consists of three LiDARs 2D, IMU and wheel odometry sensors. The proposed simulation-based methodology is a cost-effective alternative to real-world evaluation. It allows easy automation and provides access to precise ground truth. It is especially beneficial in the early stages of product design and to reduce the number of necessary real-life tests and hardware configurations.

1 Introduction

One of the most challenging topics in Robotics and Computer Vision is to enable autonomous robots and vehicles to navigate in unknown complex environments. To make it possible it is necessary to build a map of such environments and simultaneously determine robot's or vehicle's location within the created map. This procedure is called Simultaneous Localization and Mapping (SLAM), which is one of the core tasks of an autonomous robot or vehicle, as many different applications strongly depend on the generated maps².

Autonomous systems operation is possible due to data gathered from multiple embedded sensors, mainly, LiDARs (Light Detection And Ranging), radars, cameras, odometry sensors, IMUs (Inertial Measurement Unit), and GNSS (Global Navigation Satellite System). Information gathered using these sensors is utilized in the four main components of autonomous systems (localization and mapping, understanding of the surrounding environment, path determination and vehicle control).

Safety of autonomous vehicles is crucial, and to achieve it, exact position and orientation are necessary, thus localization is one of the most important aspects of the autonomous systems. For localization, GNSS is often used, however it requires a GNSS receiver with an unobstructed line of sight to minimum four GNSS satellites to work properly³. Even high-end GNSS-based systems suffer from multi-path interferences (as a results localization of the vehicle can jump up to a few meters)³. In the domain of Simultaneous Localization and Mapping, indoor environments are a major challenge, because GNSS cannot be used to obtain an absolute position due to Radio Frequency signal blocking. Therefore, other technologies are used as a basis of SLAM system for indoor environments.

In practical applications, LiDARs have advanced from numerous other sensors and technologies that were not sufficiently sensitive and accurate⁴. LiDARs determine distance to other objects by emitting laser beams and measuring their travelling time. Based on these readings, clouds of points are created that represent the surrounding environment. LiDARs offer broader field of view, as well as higher resolution than radars and ultrasonic sensors. Their operation is also more robust under different conditions, including light and dark environments, with and without glare and shadows⁵. This is crucial for vehicles and robots working in demanding environments, e.g. underground ones. Also, the closer spacing of light beams results in LiDARs having better angular and temporal resolutions than Radars⁴. In opposite to camera-based sensors, they offer more robustness and less

noisy data. Also, they are less sensitive to changes in the lighting. All of these features make LiDARs the most reliable data source for the SLAM algorithm input⁶.

SLAM approaches can also be based on a fusion of data from LiDARs and different sensors, e.g. IMUs (Inertial Measurement Unit) and odometry sensors. These approaches aim to improve SLAM accuracy by estimating their position relative to a starting point. IMUs are often used in SLAM algorithms to provide an initial pose estimation of the robot (this initial pose is used for the scan matching problem), which significantly improves accuracy and stability of data association in dynamic environments⁷. IMUs, in contrast to GNSS systems do not rely on external information sources (which can be blocked or disturbed) and can provide information such as velocity and position based on the accelerometer and gyroscope readings over time. Nevertheless, inertial sensors suffer from drifts, therefore localization systems based on IMU data are subject to a rapid degradation of position over time³.

Also, data from other on-board sensors can be used to estimate changes in orientation and position relative to an initial location of the robot - e.g. wheel odometry sensors. In the case of wheeled vehicles, odometry is based on the changes in the movement of wheels. Here, rotation optical encoders or the ones based on Hall effect can be used and knowing the diameter of a wheel, its approximate linear displacement can be calculated. Based on wheels' translation and the distance between the wheels, pose of the robot can be obtained. Also, the rotation angle of each wheel is calculated based on the encoder data in real time. Unfortunately, wheel odometry usually suffers from errors due to the integration over time resulting in the final pose estimation accuracy being poor and noisy, thus other sensors have to be used to obtain an accurate localization system⁸.

Aside from input data, many different approaches have been proposed to solve the Localization and Mapping problem. These approaches are usually most of which can be categorized into the following two methods: filtering-based and optimization-based ones². The examples of filtering-based approaches are Extended Kalman filter or particle filters. They are used due to the fact that sensor data suffers from inconsistency and noise and these approaches allow modeling of different noisy sources and how they affect the measurements. The second group of methods - optimization-based one gained much popularity due to their effectiveness, robustness, scalability and better stability than the one of filtering-based approaches². In such approaches measurements are usually represented in a form of a graph, which nodes represent poses of a robot and edges represent spatial constraints between different poses².

One of the optimization-based approaches is Google Cartographer. It is one of the leading SLAM algorithms, which is compatible with Robotic Operating System (ROS) - a commonly used system in the field of robotics⁹. Cartographer can operate based on 2-dimensional and 3-dimensional LiDAR point clouds. The system is based on two types of SLAM: local and global SLAM. Local SLAM is responsible for matching the scans, incorporating adjustments regarding motion (angular and linear), creation of sub-maps and determining current trajectory¹⁰. Global SLAM, on the other hand, executes sparse pose adjustment, matches these created sub-maps and then produces a global map¹¹. It is also used to detect loop closure - a common issue in the autonomous driving area - and to correct the global map in the case of detection¹². As a result of the algorithm, the coordinates of the vehicle and the surrounding map are produced and sent to a navigation algorithm.

In this work, we present the cost-efficient evaluation methodology that can be used to test and compare different SLAM algorithms based on data from LiDARs, IMU and odometry. For that purpose, we utilize a simulated environment introduced in¹, that was comprehensively evaluated based on measurements obtained with simulated and actual devices in real-life scenes. It was proven that the proposed framework delivers very accurate and realistic data. In this article, we use this simulator to compare different hardware configurations of Google Cartographer SLAM algorithm (we use different subsets of the following devices: LiDARs 2D and 3D, IMU and wheel odometry). We compare the performance of Cartographer operating with different hardware configurations based on the accuracy of generated maps, localization errors and stability. Evaluation of different hardware configurations in simulation results in cost-efficient and more robust planning and evaluation of real world autonomous robots and systems without a need of testing them in real-life environments (at least in the early stages of development). Moreover, compatibility of a simulation with ROS allows parameter tuning and easy deployment of evaluated configurations in actual devices.

2 Related work

The topic of Indoor mapping and localization has attracted considerable interest in recent years. Thus, research papers can be found, which compare different SLAM methods and systems in terms of mobile robotics indoor navigation and mapping technologies. In¹³ they compare four ROS-based monocular SLAM methods: REMODE, ORB-SLAM, LSD-SLAM and DPPTAM. It was proven there that these visual SLAM algorithms can successfully detect large objects, obstacles and corners, however they had difficulties with detection of homogeneously colored walls (common in indoor environments), which strongly limits applicability of monocular SLAM-related techniques for indoor mapping and navigation. In¹⁴ different ROS-based visual SLAM algorithms were tested. They compared the obtained trajectories using data from different sensors: a traditional camera, a LiDAR, a stereo camera and a depth sensor. Trajectories were determined by monocular ORB-SLAM and DPPTAM,

RTAB-Map and stereo ZedFu. Again, it was proven that visual SLAMs have problems with flat monochrome objects detection. Other papers dealing with visual SLAM comparison are^{15–17}.

Other works focus only on LiDAR-based SLAM algorithms comparison, e.g.^{18–23} In¹⁸ they compared the following 2D ROS-based SLAM techniques: CRSM SLAM, Gmapping and Hector SLAM on a Roomba 645 robotic platform. They used RGB-D Kinect sensor to emulate a laser scanner. In¹⁹ they conducted experiments with : Gmapping, HectorSLAM, KartoSLAM, LagoSLAMa and CoreSLAM. They compared these algorithms in 2D simple simulations (in ROS Stage) and real world experiments. The simulation assumed no noise added to sensor data and a simple environment with a few walls only. They discussed strengths and weaknesses of each tested solution. In²⁰ they compared the following 2D SLAM libraries available in ROS: Gmapping, Google Cartographer and Hector SLAM. In this work, maps constructed using these algorithms were evaluated against the precise ground truth obtained by laser tracker in a static indoor space based on average distance to the nearest neighbor. The obtained results have shown that almost in all cases Google Cartographer obtained the smallest error while generating maps relative to the ground truth presented by a laser tracker. In²² they compared the performance of three mapping systems (Matterport, SLAMMER and NAVIS) in two different indoor environments and provided the discussion on advantages and disadvantages of these systems. In²³ they compared both the LiDAR-based and monocular camera-based SLAM algorithms. Regarding 2D lidar-based algorithms, they tested GMapping, Hector SLAM and Google Cartographer. The results have shown that out of LiDAR-based systems, Google Cartographer demonstrated the best performance and the biggest robustness to environmental changes.

Papers present in the literature regarding the comparison of SLAM algorithms include comparative studies of different SLAM systems (here the necessity to purchase several entire hardware platforms occurs). The advantage of such works is rather impartial evaluation of off-the-shelf solutions for mapping indoor environments. On the other hand, the disadvantage is that there is often a need to add an autonomy system to an already existing robot, rather than providing a ready-made system. Other works compare different SLAM algorithms, making it often necessary to build different hardware platforms and purchase a large number of sensors, as the solutions are compatible with different physical devices. Their advantage is the evaluation of the effectiveness of different algorithms under the same or similar conditions. However, a big disadvantage is that this type of approach will not work when, for example, a set of possible sensors is given in advance and the aim is to test an optimal subset of them. Our solution focuses on creating a method to test and evaluate the performance of the SLAM algorithms using different sets of sensors (2D and 3D Lidars, IMUs, wheel odometry sensors) to be mounted on a real room decontamination robot. SLAM algorithm taken into consideration is Google Cartographer. This SLAM algorithm turned out to be the best one among the tested algorithms in the 2 papers described above, so the analysis of its different hardware configurations is valuable and, as far as we know, has not been previously covered in the literature.

Papers regarding the usage of simulation for LiDAR-based systems evaluation are limited. In²¹ they used Gazebo simulator to test the performance of navigation of an autonomous golf cart. They deployed numerous sensors in their model (reflecting the actual vehicle): LiDARs, GPS, Camera, IMU, wheel encoders and sonars. Although this use case regards the outdoor environment, it is one of the few works using more realistic simulation for performance evaluation (than e.g. ROS Stage), in this sense it is similar to our approach. Gazebo provides simplified sensor data without taking noise into account and allows very limited control over the operation of these components (which makes tuning parameters using simulation impossible). Although, Gazebo can undoubtedly be used for preliminary tests of robot localization and navigation, it cannot be used for comprehensive evaluation of SLAM algorithm performance. The discussion regarding the usage of different simulation platforms for the purpose of SLAM algorithms performance evaluation can be found in¹. These platforms (CARLA²⁴, AirSim²⁵ and LiDARsim²⁶) focus mainly on realistic scene generation and data labeling for the purpose of object recognition and not on the creation of realistic raw sensor data. They also consider mainly outdoor, urban environments for the purpose of autonomous cars testing and not the indoor environments (hospitals, airports and supermarkets), in which Location-Based Services are also crucial.

3 System Overview

3.1 Real system to be modeled

The robot considered in this paper is a remotely controlled robot designed for decontamination of indoor spaces (e.g. hospital rooms). The main task was to develop an autonomy system for this robot, its evaluation and future deployment. The robot uses powerful UV-C lamps, which aim to neutralize viruses (including SARS-CoV 2), bacteria and other microorganisms. By matching the power of the UV lamps and the speed of the robot, surface decontamination can be carried out more efficiently and safely while maintaining high performance.

The implementation of the autonomous driving system for decontamination began with an analysis of the requirements and environmental conditions of the robot. Operating in narrow corridors, among beds in hospital rooms or around chairs in office spaces requires high accuracy and precision of the vehicle localization and mapping system, and also introduces constraints on

the size and shape of the robot. In order to ensure high performance and accuracy of the system, it was decided to test different configurations of sensors mounted on the robot. The following devices were considered:

- three low-cost 2D LiDARs positioned to achieve a 360 degree field of view
- one 3D LiDAR positioned on top of the robot
- high rate IMU
- encoders placed in the BLDC motors that control the wheels of the robot, allowing the calculation of the odometry.

In Table 1 the parameters of these sensors have been summarized.

LiDAR 2D RPLIDAR A2M8	
Range	12 m
Wavelength	785 nm
Field of View	360 deg
Angular resolution	0.9 deg
Range error	<1% of the distance
Max. measurement rate	8k points/s
Scan rate	10 Hz
Sample duration	0.125 ms
LiDAR 3D Velodyne VLP-16	
Range	100 m
Wavelength	903 nm
Field of View (Horizontal)	360 deg
Field of View (Vertical)	30 deg
Channels	16
Angular resolution (Horizontal)	0.25 deg
Angular resolution (Vertical)	2.0 deg
Range accuracy	+/- 3 cm
Max. measurement rate	8k points/s
Scan rate	10 Hz
Sample (1 firing sequence - all 16 lasers) duration	55.296 us
IMU	
Gyroscope/Accelerometer Data Rate	100Hz
Gyroscope RMS noise	0.06 deg/s
Accelerometer RMS noise	0.06 m/s ²
Gyroscope Dynamic Range	+/- 2000 deg/s
Accelerometer Dynamic Range	+/- 8 g
BLDC motor	
Voltage	24V
Power	250W
Control	Three-phase, trapezoidal commutation
Encoder	Hall Effect Sensor

Table 1. Technical specification of actual sensors used in the study

The basis of the experiments was a system with three 2D LiDARs, due to the relatively low cost of these devices compared to 3D LiDARs (which is extremely important for future production of the system under consideration and other practical solutions, both the scientific and the commercial ones). In practical solutions it should be taken into account that adding more sensors involves costs, so available configurations should be analysed in order to select the optimal one in terms of performance and number/types of sensors. Performing this type of analysis in a highly realistic simulation means that such an analysis can be carried out without having to purchase all the components in the early testing stages. In addition, by creating sensor models that correspond to their real-life counterparts, it is possible to tune the parameters of autonomy algorithms and the operation of these sensors in simulation, which can significantly accelerate and facilitate the development and deployment process.

3.2 Google Cartographer SLAM algorithm

Google Cartographer is a system responsible for simultaneous localization and mapping in both 2D and 3D and also across multiple platforms and different sensor configurations¹⁰. It is a platform that is suitable also for systems with limited computational resources. Its main source of data are LiDARs (both 2D and 3D LiDARs can be used). Other data can be added to potentially improve the SLAM accuracy: odometry pose, IMU data (which delivers information regarding angular velocity and linear acceleration) and fixed frame pose. Two types of SLAM are used by the system: Local and Global SLAM¹⁰. Local SLAM is responsible for scan matching (optimization with Ceres Solver²⁷), motion filtering, sub-maps and current trajectory creation. The information generated by Local SLAM goes to Global SLAM, in which the following tasks are performed: matching the sub-maps generated by Local SLAM and creation of a global map¹¹, loop closure detection and correction of a global map when loop closure is detected¹².

To build the map, Cartographer uses the projection of successive laser scans onto a 2D plane. To effectively estimate the position of successive scans, especially in the case of unstable platforms²⁸, IMU is used to estimate the gravity vector. Furthermore, based on the angular velocity measured by the gyroscope embedded in IMU, it is possible to estimate the rotation value between successive lidar scans. In the absence of IMU, it can also be obtained using a more rigorous and accurate scan matcher at the expense of the computational complexity of the SLAM algorithm²⁸. However, the computational power requirements for 3D lidar usage, make it mandatory to use IMU to properly determine the position between successive scans.

3.3 Modeling of the components and simulation

To achieve a high-quality simulation, it was necessary to create a highly accurate model of the robot (see Figure 1) and all the sensors under test. Other available simulations do not place such a priority on accurate simulation of the raw sensor data, which is crucial for meaningful evaluation of the SLAM system in the simulation. Not taking sensor-specific phenomena and errors into account may (very likely) result in the inability to perform an objective evaluation in the simulation. Therefore, in our simulation we put great emphasis on the correct simulation of data generated by 2D and 3D LiDARs (our LiDAR simulation is described in detail in the article¹, in which we describe the effects present in real lidar data, for example rolling shutter effect, and comprehensively evaluate the accuracy of this simulation), IMU and wheel encoders.

Figure 1. A screenshot from the simulation environment presenting the modeled robot

Values that describe sensor-specific quantities along with the measurement time are sent via a predetermined protocol to ROS, where they are packed into standard ROS messages of type `sensor_msgs/sensor_type`. The data is sent in the same way for IMU, odometry and laser scan data. These data samples are then subscribed by the ROS node Cartographer, which executes the SLAM algorithm. In the case of actual 3D LiDARs, e.g. Velodyne LiDARs, they deliver data in small packets (and not the whole pointcloud at once). Cartographer takes these small packets as input and utilizes them to build a full 360 degree scan by calculating the translation of the smaller parts of the scan based on IMU data. That is why when LiDARs 3D are used as

scanners, IMU sensor has to be used as well. Due to the high frequency of simulated data generation, the LiDAR frames are accumulated (parameter `num_accumulated_range_data`) in a similar way as it happens real life and combined into a one larger one by the Cartographer¹.

Simulation of IMU for the purposes of integration into the SLAM algorithm requires the delivery of acceleration and angular velocity of the object in 3 axes. With the definition of acceleration as the second derivative of the path after time, the change in position of the simulation object between two points in the time period corresponding to the frequency of the real sensor is determined, and then, based on the determined velocity and the previous value, the acceleration is calculated. The final step is to take into consideration the value of the ground acceleration. Similarly, the angular velocity of the object is determined, i.e. the change in the distance travelled in a given time segment is utilized. This is supplemented by the quaternion of the object's rotation, which is read directly from the simulation. Then, all determined values must be burdened with an error, the value of which is a pseudo-random number from the range of values provided by the manufacturer of a given sensor model or any that can be determined by empirical testing of a real device.

The odometry data used for the SLAM algorithm in Cartographer consists of two components: a pose and a velocity vector, and the corresponding covariance matrices. These values can be calculated, for example, from input data from wheel encoders that count motor revolutions. In our case, each of the two BLDC motors embedded in the wheel is equipped with Hall sensors that inform about the rotor position and a digital output that signals (with the rising edge of a 5V pulse) that the wheel has rotated by 1 degree. These pulses are then counted by the motor controller using 64-bit counters, whose values provide the input for the odometry algorithm²⁹.

The simulation of odometry data can be performed at different levels of abstraction, depending on the desired results. Thus, the position and velocity vector of the vehicle can be read out from the simulator. The alternatives are the aforementioned pulse counters, or even the digital pulses themselves or data from Hall sensors monitoring the position of the rotor. In order to obtain more realistic results, we decided to simulate the encoder counters by calculating the difference in angle between the wheel rotation in successive steps of the physics engine simulation. Then, when the difference reaches a value compatible with the angular resolution of the real encoders, the counter corresponding to the wheel in question is incremented or decremented depending on the direction of movement, and the counter value itself is sent to the autonomy system at a frequency compatible with the controller used. This way, the odometry data could be calculated by the same algorithm that is used on the actual robot.

4 Materials and Methods

4.1 Experimental environment and track

We first modeled the actual laboratory room (medium-sized room with dimension $\sim 13 \times \sim 14$ m) in our simulation framework (the photo of this actual laboratory can be found in¹) to conduct the experiment in a realistic simulated environment. All of the items and furniture from the actual room were reproduced in the simulation. This way, the experimental environment takes into consideration real furniture, equipment and objects present in this type of space. Furthermore, the elements used make the test environment diverse and relatively complex. This approach minimises the risk of creating an overly simple experimental environment that would artificially enhance the results obtained and make the analysis results unreliable. The room is roughly rectangular, but not perfectly - it has some recesses. In the room there are, among others, a monitor stand, various types of laboratory machines, cabinets, tables and chairs. There are also several pillars in the room. Figure 2 presents the screenshot from the simulation and the top view.

To conduct all of the experiments, we used a model of an actual mobile robot with different hardware configurations. The robot was tasked to drive from the start point to the end point 5 times passing through each control point (the control points were located at the vertices of the square). Track is an example of a closed-loop trajectory consisting of a square. The scheduled track of the robot (here the perimeter of the square is taken into account) is the ground truth and is read directly from the simulation which greatly facilitates the measurement of the actual ground truth, which can be troublesome in the real world (it does not require the use of any additional measuring devices that themselves introduce some measurement errors). Figure 3 shows the start and end points of the track and the ground truth (the perimeter of the square) used to evaluate the localization performance.

During the ride, the task was to build a map of the surroundings and to locate and maintain a suitable track. On the basis of the results obtained, the quality of the built maps and the accuracy of the localization (based on the actual position and the ground truth discussed in the section) have been evaluated.

4.2 Evaluation of localization accuracy

Localization accuracy was assessed based on the difference between true localization of the robot (based on SLAM results) and the planned track (ground truth - the perimeter of the square), which has been already discussed shortly in Section 4.1. The track with the start and end point, as well as the ground truth marked can be seen in Figure 3.

(a)

(b)

Figure 2. Simulated laboratory room

Figure 3. View from the top of the simulation environment with the track ground truth

The comparison of the simulator position with the ground truth was performed by sending the current position determined by SLAM to the simulator, which matched the received value with the actual position reading of the robot in the simulation. SLAM computes the current position every 5ms and the current position is sent to the simulator every 100ms. When a new position arrives, the ground truth is measured in the simulation and they are both saved to a csv file with the results. Such measurements were possible due to the synchronisation of the starting position of the SLAM algorithm with the spawn point of the object in the simulation. We decided to sample the robot's position and calculate the error as frequently in order to minimize the impact of the outliers on the results, thus making them as reliable as possible.

Based on data gathered in the csv file, we calculated the following meaningful statistics for all of the examined cases: average, maximum and end value of the statistics. We also created a scatter plot of the cumulative SLAM error value as a function of consecutive checkpoints to show how the error changes over time. In Figure 4 we have presented the overview of the localization accuracy evaluation methodology.

Figure 4. Evaluation procedure of localization performance

4.3 Evaluation of mapping accuracy

To enable the accuracy evaluation of mapping performance, we used RViz to visualize maps created by Google Cartographer for all hardware configurations. To evaluate the mapping performance, we have split the problem into the following two subproblems:

- General overview and evaluation of key objects
- Accuracy evaluation of key objects representation

In the first subproblem we have focused on general features and accuracy of the occupancy grids: we have analysed the occurrence of map drift and its causes, we have overlaid the occupancy grids obtained via the SLAM algorithm on the simulation screenshots and assessed their overall accuracy in regard to the map from the simulation.

In the second stage (Accuracy evaluation of key objects representation) we have chosen some characteristic objects from our laboratory room and marked them on the simulation screenshot. In our detailed analysis we have taken into consideration the following elements:

- Walls and wall niches
- Objects close to the wall
- Free-standing objects
- Pillars

We have assessed the accuracy of representation of these objects in the occupancy grids. The objects present have been marked in Figure 5.

5 Results and Discussion

In the article we have focused on the evaluation of both key aspects of SLAM: Localization and Mapping. The accuracy evaluation of localization can be found in Section 5.1 and the one for mapping - in Section 5.2.

Figure 5. View from the top of the simulation environment with the characteristic objects marked

5.1 Comparison of localization performance

We assess the localization performance of different hardware configurations of Google Cartographer SLAM algorithm based on the scatter plots showing the cumulative SLAM error values over time (as a function of consecutive checkpoints) presented in Figure 6 and end, mean and maximum SLAM error bar plots presented in Figure 7.

5.1.1 Lidars 2D

In this case, a relatively low value of the mean SLAM error was observed, however, due to the highest value of the maximum SLAM error (caused by increasing pose estimation errors over time), the accuracy of the examined SLAM variant degrades significantly over time. It is also visible when the first moments of the experiment are analyzed: the values of the SLAM error are very similar to those of the best variants (LiDAR 2D + IMU + odometry and Lidars3D with and without odometry), and then they quickly diverge.

This degradation can be caused by the fact that, in this case, estimation is determined only on the basis of the external environment represented by LiDAR data.

5.1.2 Lidars 2D + IMU

At the beginning, the SLAM error changes faster than in the case of using LiDARs only, however after time, the increase of error in the case of LiDARs and IMU stays stable (in opposite to LiDARs only, for which the error increases rapidly at the end of the experiment). The maximum SLAM error is higher than in the case of LiDARs + odometry, however, due to much lower average error values, the final value (and the partial values) of error is much lower for LiDARs and IMU.

It shows that using IMUs can enhance the operation of LiDAR-based SLAMs over time with stability.

5.1.3 Lidars 2D + odometry

In this case, the results show the worst SLAM performance according to mean and end values of SLAM error, as well as to the figure showing the SLAM error over time (see Fig.). It is clearly visible that, although the maximum error is not the highest, the highest value of the mean SLAM error results in fast degradation of the pose accuracy estimation.

The main cause of this can be the low accuracy of odometry estimation based on data from encoders in wheels. The accuracy of these encoders is highly impacted by skidding, driving on uneven surfaces and even slight deformations of wheels.

Figure 6. Scatter plot of cumulative SLAM error over time (expressed as the consecutive checkpoints)

Figure 7. Bar plots representing: (a) end, (b) average, (c) maximum error values for different SLAM hardware configurations

All of these factors can significantly reduce the pose estimation accuracy.

5.1.4 Lidars 2D + odometry + IMU

In this case, the best results have been obtained. It is clearly visible on the figure showing the SLAM error values over time: the cumulative error value increases slowly, and the partial error values do not seem to increase over time, showing the stability of the algorithm (thus the robot) operation - the plot could be easily approximated by a straight line. The average error value for this case is the lowest of all examined cases, and the maximum value is the second lowest value (the difference between this value and the lowest one - obtained for LiDAR 3D and odometry - is only 2mm).

The results show that using data representing the inner state of the robot and the external environment allows us to achieve the highest accuracy and stability over time of the pose estimation. This configuration seems to be the best one in the case of the examined decontamination robot. It is clearly visible that the fusion of data from multiple sensors can result in better accuracy of the solution.

5.1.5 Lidar 3D and Lidar 3D + odometry

As it was already mentioned in Section 3.3, in the case of LiDAR 3D-based configuration IMU is always used, thus we compare only two 3D cases. Due to the fact, that the results for both LiDAR 3D cases were very similar, we decided to summarize them together. The results show that the LiDAR 3D without odometry in the majority of checkpoints obtained slightly better results in terms of partial cumulative error values (see Figure 6). On the other hand, the scatter plots of cumulative SLAM error values for both cases tend to converge and diverge over time. The maximum error value is lower for the LiDAR 3D with odometry.

The results obtained for LiDAR 3D cases show that the odometry utilization does not have much impact on the Cartographer performance (only maximum error values can be slightly lower when odometry is used - it might be the result of slightly better stability). The SLAM errors are less predictable when configurations with LiDARs 3D are used in opposite to the majority of 2D configurations, which manifests itself as a wavy line of scatter plots.

5.2 Comparison of mapping performance

We have used RViz to visualize maps (occupancy grids) and trajectories recovered from Google Cartographer 2D SLAM algorithm. These trajectories have already been evaluated in Section 5.1. The generated maps are assessed in terms of accuracy of detection of main scene elements (walls and key objects). To facilitate the evaluation and interpretation of the results, we not only provide the generated occupancy grids, but also the top views of the simulated environment with the occupancies grids layed on top of them (see Figures 8 and 9). We also chose some characteristic objects in the rooms to discuss the accuracy of reflection of these objects in the obtained occupancy grids (see Figure 5, which we marked and assigned the labels to 13 objects).

5.2.1 General overview and evaluation of generated maps

In Figures 8 (occupancy grids for LiDAR 2D without additional sensors, LiDAR 2D with IMU and LiDAR 2D with odometry) and 9 (LiDAR 2D with IMU and odometry and LiDAR 3D with IMU and LiDAR 3D with IMU and odometry) it is clearly visible that for all examined cases we obtained relatively accurate maps of the environment. All maps reflect the majority of main scene elements: walls, wall niches, a room behind the doorframe, pillars and the objects present in the room. It can be observed that occupancy grids obtained in the cases of LiDAR 2D with no additional sensors and LiDAR 2D with odometry suffer from drifts. In the case of SLAM using only 2D LiDARs during operation when turning, accelerating and braking it was visible how the location drifted, but when the vehicle was moving at a constant speed in a straight line the stability returned. This shows that using 2D LiDARs alone is sufficient for simple linear trajectories, but suffers from drifts with rotational motion and changes in linear velocity. Unfortunately in practical cases (mapping, navigation) such constraints are not acceptable. In the case of 2D LiDAR with odometry, map drift is much more visible and occurs when robot rotates. This may be due to inaccurate pose estimation based on yaw velocity calculations from wheel encoder data (odometry data are usually subject to large errors and noise). All the other cases use IMU data (both 2D and 3D, for the 3D case the use of IMU is mandatory): orientation, linear and angular velocity. IMU readings are usually more accurate than those from odometry, which results in the improvement of the quality of the resulting maps.

5.2.2 Accuracy evaluation of key objects representation in occupancy grids

It can be observed that all of the LiDAR 2D cases are not able to detect the walls behind the heavy machinery (see point 5 in Figure 5) in the left upper corner of maps in Figures 8 and 9. It is caused by the fact, that the 2D LiDARs due to the shape of the robot are embedded on its case (obviously, LiDARs cannot be installed on the UV lamp), thus the LiDARs are placed relatively low and they the walls under discussion wall are not in their field of view. That is why SLAM algorithm perceives heavy machinery as walls of the room in that region. In the cases with 3D SLAM walls in this region are well reflected, because the 3D LiDAR stands on top of the UV lamp (it is necessary to provide it with a 360 degree view). Nevertheless, heavy machinery in this region is less visible on the occupancy grids (a thin line). Analogous results were obtained for other objects next to the

(a)

(b)

(c)

(d)

(e)

(f)

Figure 8. Maps generated with Google Cartographer for the following hardware configurations and maps placed on the screen of the simulated environment: (a),(b) 3xLiDARs 2D; (c),(d) 3xLiDARs 2D + IMU; (e),(f) 3xLiDARs 2D + odometry

(a)

(b)

(c)

(d)

(e)

(f)

Figure 9. Maps generated with Google Cartographer for the following hardware configurations and maps placed on the screen of the simulated environment: (a),(b) 3xLiDARs 2D + IMU + odometry; (c),(d) 1xLiDAR 3D + IMU; (e),(f) 1xLiDAR 3D + IMU + odometry

walls (see points 1, 2, 7, 8 and 11 in Figure 5 and the corresponding regions of maps in Figures 8 and 9). In these cases, objects are treated as walls. One exception is for the biggest machine in the room - object 4 in Figure 5 - for this case in all of the cases we were unable to obtain a good representation of walls in this region (this object is much higher than the rest, so also in 3D cases the LiDAR was placed too low to accurately reflect this region).

Regarding free-standing objects (see points 3, 6, 10, 13 in Figure 5), they are much more visible on occupancy grids obtained with LiDAR 2D cases. Even the table (see point 6 in Figure 5), which is one of the biggest objects in the room is barely visible on occupancy grids for LiDAR 3D cases. In contrast, it is clearly marked in all of the LiDAR 2D cases. The situation is very similar for another big object - point 10 in Figure 5 - which was reconstructed very well for LiDAR 2D cases (slightly worse for LiDAR 2D with odometry) and poorly for LiDAR 3D cases. The most demanding object - point 3 in Figure 5 (a table with thin legs) - was weakly represented in the occupancy grids of all the examined cases, however for the LiDAR 2D cases it is better visible (legs of the table are visible as dots in the occupancy grid), especially for cases with IMU. The smallest object in the scene - point 13 in Figure 5 (a small cabinet) - is virtually not visible in occupancy grids of LiDAR 3D cases (a very blurry lines can be observed) and it is clearly visible for LiDAR 2D cases (the best for LiDAR 2D + IMU + odometry).

Thin pillars present in the laboratory room (points 9 and 12 in Figure 5) can be found on all occupancy grid, but similarly to other objects they are better reflected for LiDAR 2D cases than for LiDAR 3D cases, which shows that not only height (which is important in our case as 2D and 3D LiDARs are placed on different parts of the robot) of the object impacts its detection accuracy.

6 Discussion

In terms of localization accuracy, the best results were obtained for the case with LiDARs 2D, IMU and odometry and two examined 3D cases (with and without odometry). In the first case, the advantage seems to be usage of external (LiDARs) and internal (IMU, odometry) information to perform the localization and mapping. In the second case, using odometry does not have a significant impact on the accuracy.

In terms of mapping performance, for the 3D cases, the walls of the room were better represented, and for the 2D cases - the objects. It is clearly visible, that the accuracy of wall reconstruction and object detection is highly dependent on laser scanner/scanners placement on the examined robot (2D scanners were placed on the case of the robot and 3D scanner - on the top of the UV lamp, which was the only possible place due to the shape of the robot). Therefore, in practical applications sometimes it may be impossible to find the perfect sensor placement that would reflect all of the objects in the room.

When it comes to additional sensors, the use of IMU allows to improve the driving stability of the vehicle (which results in low SLAM error values) and, consequently, to obtain a map of significantly better quality. Using odometry can lead to decrease in Simultaneous Localization and Mapping accuracy, however when used along with IMU sensors can deliver the highest quality results.

The great advantage of testing the system in the design phase via simulation is that there is no need to purchase sensors to test all the possible configurations and it is possible to automate this process, as well as to obtain identical conditions for each configuration tested. Below, we provide the list of advantages and disadvantages of testing different algorithms and hardware configurations in a simulation.

Advantages of SLAM algorithm testing in simulation:

- Possibility to perform more robust tests and more tests in general - it is possible to easily automate SLAM accuracy testing against ground truth in a large number of measurement points (which in reality involves laborious and troublesome measurements burdened with measurement error)
- Access to precise ground truth
- Possibility of conducting experiments under perfectly identical conditions in real time (repeatability of tests)
- Either a map corresponding to a real room can be created or your resources can be easily increased - for example by creating a huge production area in the simulation - it is particularly valuable in the earlier stages of manufacturing
- Cost-effectiveness (only a narrowed down set of sensors has to be purchased)

Disadvantages of SLAM algorithm testing in simulation:

- Building a very accurate virtual environment is time-consuming or impossible. Nevertheless, in some cases it may not be required.

- Modern simulations (especially those modeled in advanced game engines) are very accurate, but we are still dealing with simplified physics, world and sensors' models. Still in the later stages of production it would be beneficial to invest in real hardware and perform tests (however, using simulation we can significantly reduce the number of necessary tests)

This list can be used in the decision-making process regarding how to test a production solution and whether simulation is suitable for a particular application. Generally, it seems to be especially helpful in the early stages of product design and to reduce the number of necessary real-life tests.

7 Conclusions

In the article, we have presented the evaluation methodology that can be used to test and compare different SLAM algorithms based on data from LiDARs, IMU and odometry. We have also discussed the advantages and disadvantages of such an evaluation strategy.

In our case, we have used a simulated environment introduced in¹ to compare different hardware configurations of Google Cartographer SLAM algorithm to be deployed in an actual robot used for decontamination (thus the possible placement of different sensors was limited). We examined six different hardware configurations based on 3 LiDARs 2D, one LiDAR 3D, IMU and wheel odometry. We have compared these configuration in terms of localization accuracy (measured based on SLAM localization error) and mapping performance (based on generated maps analysis regarding the occurrence of key objects representations in occupancy grids and general accuracy of generated maps, i.e. shape of the examined room etc.).

The results of our analysis have shown that in our case the hardware configuration consisting of three LiDARs 2D, IMU and wheel odometry is the best choice in terms of localization accuracy and mapping performance. Due to the placement of the LiDARs at the bottom of the robot (it was the most suitable place regarding the robot's shape), some walls of the room have not been mapped properly, as they were simply obscured by tall equipment. Nevertheless, the overall mapping accuracy was very good and sufficient for the application. Moreover, 3D LiDARs had to be placed on the top of the robot, therefore some of the lower object were not detected (or virtually not visible), which can decrease the applicability of the robot in the indoor environments (e.g. the robot can drive into such an object). This shows that 3d LiDARs would be better for lower robots.

Simulation-based testing allows performing more robust test and more tests in general (as it can be easily automatize), it provides access to precise ground truth, perfectly identical testing conditions in real time and can be used to model larger environments than ones we have access to. Generally, it is a cost-effective alternative to real-world evaluation, especially beneficial in the early stages of product design and to reduce the number of necessary real-life tests.

References

1. Sobczak, Ł., Filus, K., Domański, A. & Domańska, J. Lidar point cloud generation for slam algorithm evaluation. *Sensors* **21**, 3313 (2021).
2. Doaa, M., Mohammed, A., Salem, M., Ramadan, H. & Roushdy, M. I. Comparison of optimization techniques for 3d graph-based slam. *Recent Adv. Inf. Sci.* (2013).
3. Meng, X., Wang, H. & Liu, B. A robust vehicle localization approach based on gnss/imu/dmi/lidar sensor fusion for autonomous vehicles. *Sensors* **17**, 2140 (2017).
4. Khan, M. U. *et al.* A comparative survey of lidar-slam and lidar based sensor technologies. In *2021 Mohammad Ali Jinnah University International Conference on Computing (MAJICC)*, 1–8 (IEEE, 2021).
5. Yue, X., Wu, B., Seshia, S. A., Keutzer, K. & Sangiovanni-Vincentelli, A. L. A lidar point cloud generator: from a virtual world to autonomous driving. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, 458–464 (2018).
6. Ren, Z., Wang, L. & Bi, L. Robust gicp-based 3d lidar slam for underground mining environment. *Sensors* **19**, 2915 (2019).
7. Wang, Z., Chen, Y., Mei, Y., Yang, K. & Cai, B. Imu-assisted 2d slam method for low-texture and dynamic environments. *Appl. Sci.* **8**, 2534 (2018).
8. Toledo, J., Piñeiro, J. D., Arnay, R., Acosta, D. & Acosta, L. Improving odometric accuracy for an autonomous electric cart. *Sensors* **18**, 200 (2018).
9. Nüchter, A., Bleier, M., Schauer, J. & Janotta, P. Improving google's cartographer 3d mapping by continuous-time slam. *The Int. Arch. Photogramm. Remote. Sens. Spatial Inf. Sci.* **42**, 543 (2017).
10. Cartographer — Cartographer documentation. [online]. Available: <https://google-cartographer.readthedocs.io/> [Accessed: 2021-10-11].

11. Bailey, T. & Durrant-Whyte, H. Simultaneous localization and mapping (slam): Part ii. *IEEE robotics & automation magazine* **13**, 108–117 (2006).
12. Ji, X., Zuo, L., Zhang, C. & Liu, Y. Lloam: Lidar odometry and mapping with loop-closure detection based correction. In *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, 2475–2480 (IEEE, 2019).
13. Buyval, A., Afanasyev, I. & Magid, E. Comparative analysis of ros-based monocular slam methods for indoor navigation. In *Ninth International Conference on Machine Vision (ICMV 2016)*, vol. 10341, 103411K (International Society for Optics and Photonics, 2017).
14. Ibragimov, I. Z. & Afanasyev, I. M. Comparison of ros-based visual slam methods in homogeneous indoor environment. In *2017 14th Workshop on Positioning, Navigation and Communications (WPNC)*, 1–6 (IEEE, 2017).
15. Sokolov, M., Bulichev, O. & Afanasyev, I. Analysis of ros-based visual and lidar odometry for a teleoperated crawler-type robot in indoor environment. In *ICINCO* (2), 316–321 (2017).
16. Giubilato, R., Chiodini, S., Pertile, M. & Debei, S. An experimental comparison of ros-compatible stereo visual slam methods for planetary rovers. In *2018 5th IEEE International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, 386–391 (IEEE, 2018).
17. Mingachev, E. *et al.* Comparison of ros-based monocular visual slam methods: Dso, ldso, orb-slam2 and dynaslam. In *International Conference on Interactive Collaborative Robotics*, 222–233 (Springer, 2020).
18. Rojas-Fernández, M., Mújica-Vargas, D., Matuz-Cruz, M. & López-Borreguero, D. Performance comparison of 2d slam techniques available in ros using a differential drive robot. In *2018 International Conference on Electronics, Communications and Computers (CONIELECOMP)*, 50–58 (IEEE, 2018).
19. Santos, J. M., Portugal, D. & Rocha, R. P. An evaluation of 2d slam techniques available in robot operating system. In *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 1–6 (IEEE, 2013).
20. Yagfarov, R., Ivanou, M. & Afanasyev, I. Map comparison of lidar-based 2d slam algorithms using precise ground truth. In *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 1979–1983 (IEEE, 2018).
21. Shimchik, I., Sagitov, A., Afanasyev, I., Matsuno, F. & Magid, E. Golf cart prototype development and navigation simulation using ros and gazebo. In *MATEC Web of Conferences*, vol. 75, 09005 (EDP Sciences, 2016).
22. Chen, Y. *et al.* The accuracy comparison of three simultaneous localization and mapping (slam)-based indoor mapping technologies. *Sensors* **18**, 3228 (2018).
23. Filipenko, M. & Afanasyev, I. Comparison of various slam systems for mobile robot in an indoor environment. In *2018 International Conference on Intelligent Systems (IS)*, 400–407 (IEEE, 2018).
24. Dosovitskiy, A., Ros, G., Codevilla, F., López, A. & Koltun, V. Carla: An open urban driving simulator. In *CoRL* (2017).
25. Shah, S., Dey, D., Lovett, C. & Kapoor, A. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and service robotics*, 621–635 (Springer, 2018).
26. Manivasagam, S. *et al.* Lidarsim: Realistic lidar simulation by leveraging the real world (2020). [2006.09348](https://arxiv.org/abs/2006.09348).
27. Ceres Solver — A Large Scale Non-linear Optimization Library. [online]. Available: <http://ceres-solver.org/> [Accessed: 2021-10-11].
28. Hess, W., Kohler, D., Rapp, H. & Andor, D. Real-time loop closure in 2d lidar slam. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 1271–1278 (IEEE, 2016).
29. CleanIt - Odometry Algorithm. [online]. Available: <https://github.com/Sollimann/CleanIt/tree/main/autonomy/src/slam/odometry/> [Accessed: 2021-10-11].

Author contributions statement

Ł.S. and K.F. conceived the experiments and prepared the original draft, Ł.S. conducted the experiments, J.D. and A.D. analysed the results. All authors reviewed the manuscript.