Received: January 15, 2023. Revised: February 27, 2023.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

An Extensive Analysis and Fine-Tuning of Gmapping's Initialization Parameters

Zuhair A. Ahmed[1]* **Safanah M. Raafat[1]**

[1]*University of Technology, Iraq*
* Corresponding author's Email: cse.20.27@grad.uotechnology.edu.iq

Abstract: Accurate localization and mapping are essential for autonomous navigation systems. The simultaneous localization and mapping (SLAM) algorithm, gmapping, is widely used for creating two-dimensional occupancy grid maps (2D-OGMs) due to its low cost and effectiveness in indoor environments. According to the SLAM interaction between localization and mapping terms, the evaluation of map accuracy implicitly means the evaluation of localization. This study focuses on improving the accuracy of the estimated 2D-OGMs by fine-tuning the 32 initialization parameters of gmapping using a turtlebot3-burger robot. An improved experimental procedure was developed, incorporating image registration and similarity measurement to evaluate the map accuracy. The results show a substantial improvement in map accuracy, from 78.84% to 94.18%. The study highlights the importance of fine-tuning the SLAM algorithm for improved map accuracy and provides valuable insights for developing autonomous navigation systems. The key contribution of this paper lies in the systematic classification, fine-tuning, and evaluation of the Gmapping initialization parameters.

Keywords: Mobile robot, Autonomous navigation, Simultaneous localization and mapping (SLAM), Gmapping, fine-tuning, Occupancy grid mapping.

1. Introduction

Mobile robotics has made significant advancements in recent years, with applications in various fields, including but not limited to surveillance, emergency response, industrial automation, and personal services [1]. In order to achieve effective navigation, mobile robots must have an accurate and up-to-date map, localize their position, plan their motion, and identify and avoid potential obstacles [2].

One approach to this is through SLAM algorithms, which allow the robot to create a map of its surroundings and simultaneously estimate its pose within that map [3]. Over the last few years, numerous algorithms for solving navigational problems have been developed. Most of these are available in robot operating system (ROS), which this work will use as a meta-operating system [4].

According to the algorithmic principle, There are two main categories of probabilistic SLAM techniques: Bayes filters (such as Kalman filters and particle filters) and graph-based algorithms [5]. This research focuses on the former category, the particle filter-based approach, which involves a two-step process of prediction and update to improve the accuracy of the system's state estimation [6].

Gmapping is a popular choice for generating 2D-OGMs using two-dimensional light detection and ranging (2D-LiDAR) sensors due to its low cost and effectiveness [7]. Moreover, its map accuracy can be improved by fine-tuning its initialization parameters [8].

Previous works have fine-tuned specific parameters of Gmapping. In [9], the authors introduced two methods to fine-tune 11 of 32 Gmapping parameters. First, shuffle parameters separately by multiplying their values by a specific factor. Second, fine-tune parameters collectively, and the final results are evaluated visually. In [10], only two parameters were tuned in hybrid indoor and outdoor environment scenes, and the evaluation of CPU and Memory usage and visual inspection was presented. Other preceding works, such as [9-14], have tuned several specific parameters without methodology to study and categorize all parameters functionality of Gmapping. No scale-defined ground truth map for the evaluation or reliable comparison has been made.

A metric for comparing SLAM algorithms was developed in [15], where the outcome is not evaluated using a reference but rather by considering the robot's poses during data acquisition. However, evaluating the estimated map quality is currently done by determining the difference between a reference map and the map under scrutiny. The structural similarity index measure (SSIM) and the iterative closest point (ICP) each of them has been used in [16] as a method to compare the similarity of the estimated maps. The K-nearest neighbor was used In [17] to evaluate OGM's accuracy as a map similarity metric for indoor and outdoor environments and CPU and memory performance evaluation. The average distance to the nearest neighbor (ADNN) was used in [18] to evaluate the mapping results of each SLAM technique for a simulation-based environment with an examination of computational load.

However, previous works had only tested SLAMs in limited scenarios in simulation or experimentally based environments where the robot was teleoperated with limited velocity implying that circumstances are nearly ideal. In addition, no systematic methodology for categorizing or fine-tuning the initialization parameters has been proposed to improve the performance of Gmapping and its estimated map.

Based on that, in this work, a systematic categorization via functionality, systematic fine-tuning, and an evaluation metric of the estimated map are all expanded to include scenarios in which a robot engages in the autonomous exploration of realistic indoor environments. All were done experimentally under experimental processes and realistic constraints, such as drifting caused by velocity variety, motion model noise, and a real room-scale closed environment. The objectives of this work are to:

- Analyze and categorize all the initialization parameters of the Gmapping SLAM algorithm due to functionality.
- Adapt the effective parameters for a realistic indoor environment using turtlebot3 burger-type (TB3B) by testing the accuracy of the generated grid map via ORB and SSIM techniques [16].
- Specify the final optimal parameters by testing under another locomotion and environment specifications.

The rest of this paper is structured as follows. In sect. 2, the system overview consists of a probabilistic RBPF-based SLAM and Gmapping processing model. The experimental design section includes the robot model, trajectory scenario, environmental specifications, experimental procedure, and classification of parameters into groups. Sec. 4 the results and discussions about the tuning and evaluation of each group. Finally, the conclusions are demonstrated in sect. 5.

2. System overview

The probabilistic approaches intended to include uncertainties in the system model. Particle filters (PF), a variant of recursive state estimator based on Monte-Carlo techniques, estimate the probability distribution of the system's state by sampling hypotheses. Unlike parametric filters such as KF, Extended-KF, or Unscented-KF, PF can be used for non-parametric probability distributions [19].

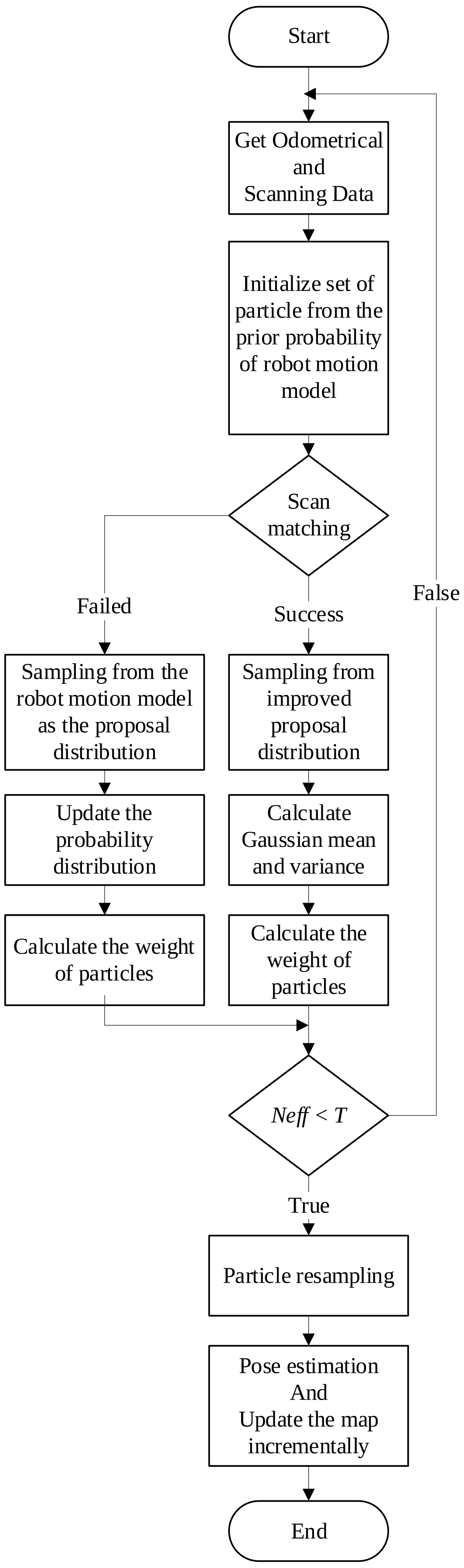
Rao-blackwellized PF (RBPF) has been developed to address the complexity of the SLAM problem, which factorizes the problem into two parts: the non-Gaussian posterior of positioning and the posterior probability estimation of the map [20], as shown in Eq. (1).

$$\underbrace{p(x_{1:t}, m|u_{1:t}, z_{1:t})}_{SLAM} = \underbrace{p(x_{1:t}|u_{1:t}, z_{1:t})}_{Localization}\underbrace{p(m|x_{1:t}, z_{1:t})}_{Mapping} \quad (1)$$

In this equation, $p(x_{1:t}|u_{1:t}, z_{1:t})$ represents the robot's pose, given measurements ($z_{1:t}$) And odometry data ($u_{1:t}$), i.e., a pure localization problem. The mapping problem, $p(m|x_{1:t}, z_{1:t})$, refers to estimating the map given the robot's pose and scan measurements. One type of map commonly used in SLAM algorithms is an occupancy grid map (OGM), which is a discretized metric map represented as a grid of cells, each of which can be classified as occupied, free, or unknown based on the probability of the cell's navigability [21].

While RBPF successfully addresses the SLAM problem, it has disadvantages, such as a high number of particles and frequent resampling. Gmapping, one of the most widely used 2D-LiDAR grid SLAM open-source algorithms, addresses these issues by employing adaptive resampling and improved proposal distribution to reduce particle depletion. Gmapping can create indoor environment maps in real-time with low LiDAR scanning

Received: January 15, 2023. Revised: February 27, 2023.


Start
Get Odometrical and Scanning Data
Initialize set of particle from the prior probability of robot motion model
Scan matching
Failed
Success
False
Sampling from the robot motion model as the proposal distribution
Sampling from improved proposal distribution
Update the probability distribution
Calculate Gaussian mean and variance
Calculate the weight of particles
Calculate the weight of particles
Neff < T
True
Particle resampling
Pose estimation And Update the map incrementally
End
Figure. 1 Flowchart of Gmapping algorithm

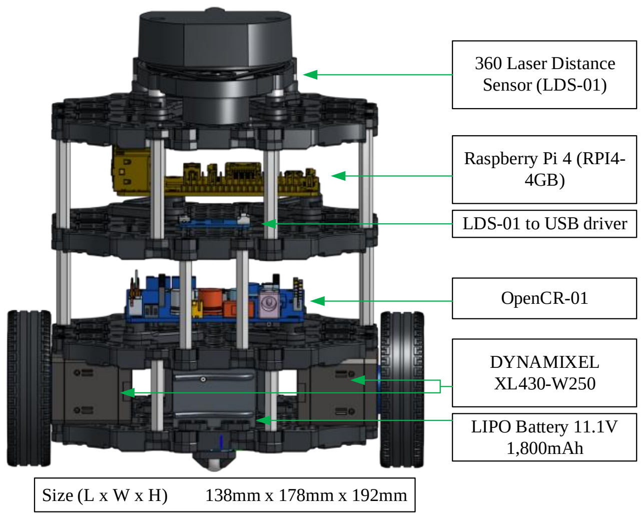

360 Laser Distance Sensor (LDS-01)
Raspberry Pi 4 (RPI4-4GB)
LDS-01 to USB driver
OpenCR-01
DYNAMIXEL XL430-W250
LIPO Battery 11.1V 1,800mAh
Size (L x W x H) 138mm x 178mm x 192mm
Figure. 2 TB3B hardware components

Table 1. Velocity variation of the mobile robot

Velocities	**Exp**	**Min**	**Max**	**Mean**	**Root Mean Square Error (RMSE)**
Linear velocity (m/s)	1st Exp	-0.080	0.200	0.03073	0.04944
Angular velocity (rad/s)		-0.700	0.900	0.00205	2.87740
Linear velocity (m/s)	2nd Exp	0.000	0.155	0.05389	0.06014
Angular velocity (rad/s)		-1.700	1.800	0.00463	2.94947

frequency needs and low computational resources while still delivering good map accuracy. The process steps of the algorithm are illustrated in the flow chart in Fig. 1. The process begins with pose prediction using the encoder and odometry data, followed by the initialization of the particle number based on the proposed motion model. Measurement and correction are then performed using LiDAR data and scan matching, followed by the weighting of samples. The final step is the map update and resampling method [22].

3. Experimental Design

3.1 Robot model

Mobile robots using differential drives are the most common in recent decades since they are the simplest form of the grounded wheeled mobile robot

Received: January 15, 2023. Revised: February 27, 2023.

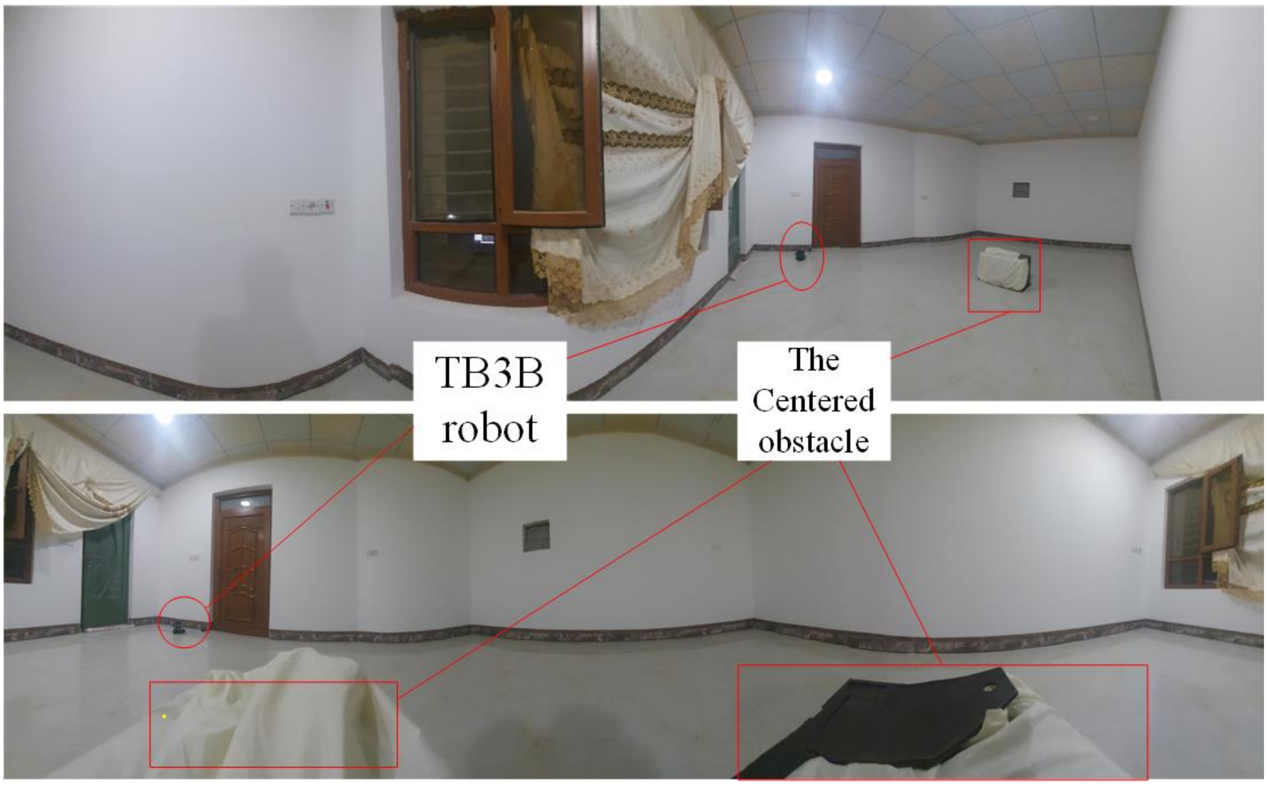


Figure. 3 Panorama images of the real environment for the 1st Exp

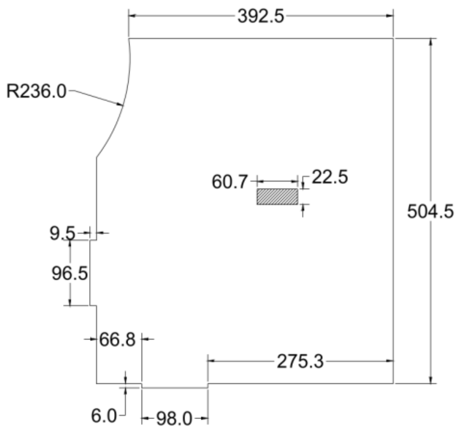
with three degrees of freedom (DoF) [23]. The differential drive term means the mobile robot can control each wheel individually. Wheel encoders and inertial measurement units provide odometer data. Highly precise LiDAR data is needed for mapping and pose correction via the scan-matching approach. When testing ROS-based 2D-LiDAR SLAM algorithms, it is best to do so in a real-world setting. TB3B model, shown in Fig. 2, is considered for realistic indoor environment mapping [24].

It is characterized by cost-effective, powerful computational features and ROS-enabled hardware components for educational and scientific purposes that can benchmark the investigated approach [25]. Therefore, it is suggested to be the mobile robot model prototype in this work.

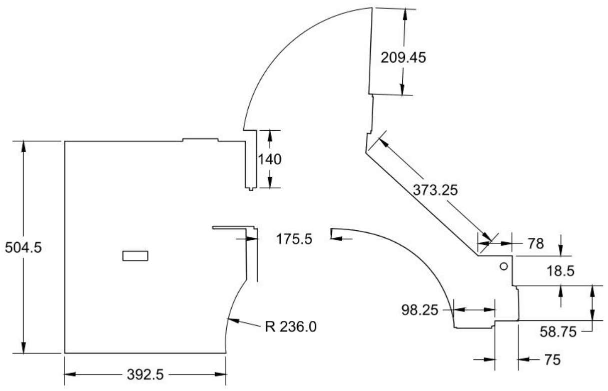
3.2 Dataset specifications

Two indoor environments are chosen for the first and second experiments (Exp)s. Both are flat terrain, room-scale, enclosed, uni-obstruction, and straightened and curved walls. All these restrictions are chosen to represent reality.

Fig. 3 shows two panorama images of the first environment. The ground truths of both environments have been measured accurately by hand once and by laser other using a distance measurement device (YT-7312250, $\pm 2\, mm$). The measurements have been projected on ground truth



(a)



(b)

Figure. 4 Ground truth maps: (a) 1st Exp ground truth map and (b) 2nd Exp ground truth map (Dimensions in cm)

Table 2. Initialization parameters of Gmapping with the default values

Parameters	Default value	Description
Group (1): Coordinate system parameters		
base_frame	Base link frame	The coordinate system of the mobile robot platform
odom_frame	Odometry frame	
map_frame	Map frame	
transform_publish_period		The publishing rate of data among frames
Group (2): OGM parameters		
xmin	-100 m	The initial map dimensions represented a rectangular region defined by two opposite corners
ymin	-100 m	
xmax	100 m	
ymax	100 m	
delta	0.05 m	The length of one side of the grid cell in meters
occ_thresh	0.25	The threshold is used to determine whether a cell is an obstacle or not.
Group (3): Measurement Update Parameters		
linearUpdate	1.0 m	The translational and rotational threshold of measurement update
angularUpdate	0.5 rad	
temporalUpdate	-1.0	The time threshold of the Measurement update
Group (4): Motion model parameters		
srr	0.1	Covariance of sampling motion noise
str	0.1	
str	0.2	
stt	0.2	
Group (5.1): Scan matching and LiDAR parameters		
maxRange	-	LiDAR maximum distance range
maxUrange	-	The usable LiDAR range
throttle_scans	1	The number of throttle scans
sigma	0.05	the Standard deviation (σ) of the score and likelihood functions of a single beam of the LiDAR
lsigma	0.075	
kernelSize	1	The size of the kernel's search window for matching scans
lstep	0.05 m	Linear and Angular displacement step increment size
astep	0.05 m	
iterations	1	The number of refinement steps used in the scan matches
lskip	0	The number of beams skipped in each scan
Group (5.2): Likelihood sampling parameters		
llsamplerange	0.01	Translation sampling range of scan matching
llsamplestep	0.01	Translation sampling step of scan matching
lasamplerange	0.005	Rotational sampling range of scan matching
lasamplestep	0.005	Rotational sampling step of scan matching
Group (6): PF parameters		
particles	30	the number of particles
map_update_interval	5.0	the map update period
minimumScore	0	The lowest score that a scan-matching algorithm must achieve for the outcome of the scan to be considered valid
ogain	3.0	the gain of weight normalizer function for smoothing the resampling step
resampleThreshold	0.5	Resampling threshold

images via AutoCAD, as shown in Fig. 4.

The first environment of Fig. 4 (a) is for the fine-tuning tests, while the second environment of Fig. 4 (b) is chosen to check the validity of the final optimal configuration. Likewise, to facilitate comparisons between the generated map and the ground truth, the applied scenario for conducting the trajectory used to build the map is tested with straight, curvature, rotation-only movement utilizing a variety of linear and angular velocities. The selected ranges of velocities are illustrated in Table 1.

3.3 Initialization parameters classification

Initialization parameters of Gmapping are critical limits of the algorithm's performance. Some are related to map resolution, particle filtering, coordinate transformation systems, scan matching, or motion model noise. In this work, parameters are classified according to theoretical reasons, functionality, and tuning considerations. Table 2 shows all the Gmapping algorithm's initialization parameters grouped into six categories.

The position of the map frame will begin from the left down corner, represented in (-xmin, -ymin) point in Group (2). Meanwhile, in Group (3), parameters such as linearUpdate, and angularUpdate, define measurement update threshold in translation (meters) and rotation (rad), respectively. On the other hand, the third parameter in Group (3) is related to time, temporalUpdate; the measurement will be performed, If the last considered measurement is older than the update time in seconds.

The fourth group is motion model parameters. Adding noise term to the motion model to describe its uncertainty makes it a probabilistic motion model, $p(x_t|x_{t-1},u_t)$, refers to the prediction step in the RBPF. According to [26], for particle filters, sampling from a conditional density is more accessible to implement and faster than calculating the density by the closed-form algorithm since there is no need for an inverse model. In Gmapping, the motion model stands for associate sampling noise from zero-mean Gaussian distribution whose covariance is proportional to a linear combination of $\Delta x, \Delta y, \Delta\theta$, which is characterized by the four parameters represent pure translational (srr), translational to rotational (srt), rotational to translational (str), and pure rotational (stt) noise components as shown in Eq. (2) as a sampling function.

$$S(\sigma) = \begin{bmatrix} S\left(s_{rr}|\Delta x| + s_{tr}|\Delta\theta| + s_{xy}|\Delta y|\right) \\ S\left(s_{rr}|\Delta y| + s_{tr}|\Delta\theta| + s_{xy}|\Delta x|\right) \\ S\left(s_{tt}|\Delta\theta| + s_{rt}\sqrt{\Delta x^2 + \Delta y^2}\right) \end{bmatrix} \quad (2)$$

where ($\Delta x, \Delta y, \Delta\theta$) is the pose differences. ($s_{xy} = 0.3 s_{rr}$) by default, the term $S(\sigma)$, return a random sample from a normal distribution with a zero mean. Gmapping samples high-quality random variables from a zero mean normal distribution using the pseudo-random number generator and polar form of the Box-Muller transformation algorithm [27].

Group (5.1) is scan matching parameters. Scan matching is an algorithm to correct the pose of the robot. The default value of LiDAR maximum range distance, maxRange, is the actual characterized range subtracted by (0.01) in meters. The correspondence usable LiDAR parameter is maxUrange, where the default value is the same as maxRange. Rays beyond this range get discarded completely. The sigma parameter indicated the standard deviation (σ) of the score function of a single beam of the LiDAR. The score function Eq. (3) is an exponential decay function that decreases with the increase of (d), the distance between the pose and the single LiDAR beam reading when the endpoint is matched.

$$ScoreForSingleBeam = e^{\frac{-d^2}{\sigma}} \quad (3)$$

Another standard deviation parameter (lsigma) of the score and likelihood function of a single beam of the LiDAR decreases linearly as the difference between LiDAR beam reading and pose increases rapidly, representing the particle's weight as shown in Eq. (4).

(-1,-1)	(-1,0)	(-1,1)
(0,-1)	(0,0)	(0,1)
(1,-1)	(1,0)	(1,1)

Figure. 5 Search window of scan matching

$$LiklihoodForSingleBeam = \frac{-d^2}{\sigma} \quad (4)$$

kernelSize parameter characterizes the size of the kernel's search window for matching scans. Fig. 5 shows the visual description of a nine-square grid centered on the hit point, and neighboring unhit points are mainly utilized for the search box size when computing the score for the most likely laser beam strike in a kernel window. Parameters of Angle and linear displacement step increment size (lstep and astep) assumed the initial step size of the optimization step in the Hill-Climbing algorithm during the scan matching algorithm [28]. The Iterations parameter is the number of times the search step size changes during the optimization method in the scan matching algorithm. In Group (5.2), llsamplerange, llsamplestep, lasamplerange, and lasamplestep parameters are used as sampling steps and range of likelihood function in scan matching.

Finally, in the sixth group, the particles parameter determines the number of particles in the Gmapping algorithm. Each particle represents a possible trajectory and its map. Ogain parameter characterized the gain of weight normalizer function for smoothing the resampling step; the normalizer function is to calculate the judgment value (N_{eff}) of the resampling. resampleThreshold parameter characterizes the adaptive resampling technique in the algorithm; it is the ratio of (N_{eff}) to the particle number (N), resampling is performed if it falls below that value, as illustrated in Eq. (3).

$$N_{eff} < (resampleThreshold)N \quad (5)$$

Higher means more frequent resampling according to the evaluation formula [29] as follows in Eq. (6):

$$N_{eff} = \frac{1}{\sum_{i=1}^{N}(\widetilde{w}^i)^2} \tag{6}$$

where $\widetilde{w}^i$ is the likelihood of each particle. N_{eff} is N when there is no difference in the likelihood of particles immediately after resampling. The value of N_{eff}, decreases as the particle likelihood increases. Eventually, it approaches 1.

4. Experimental design

The evaluation metric is an image processing tool to measure the similarity between the ground truth map and the estimated map. Firstly, the image registration technique, oriented FAST and rotated BRIEF (ORB), is used to transform, align and scale the estimated image to the reference image to justify a similarity measurement [16]. The ORB algorithm combines the FAST keypoint detector and the BRIEF descriptor with several tweaks to improve performance [30]. Lastly, the mean SSIM (MSSIM) algorithm for similarity measurement checks the estimated map's accuracy [16]. The SSIM for a specific window of size (x and y) and the MSSIM for X and Y images are given by Eqs. (7) and (8), respectively:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y+c_1)(2\sigma_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_2)} \tag{7}$$

Where μ_x , μ_y: The pixel sample's mean of x and y, respectively; σ_x^2 , σ_y^2: The variance of x and y, respectively; σ_{xy}: The covariance of x and y; c_1 , c_2: variables for stabilizing the division with a weak denominator.

$$MSSIM(X,Y) = \frac{1}{M}\sum_{j=1}^{M} SSIM(x_j, y_j) \tag{8}$$

Where the X and Y represent the ground truth and the estimated map, respectively; x_j and y_j are the sub-images at the j^{th} local window: and M is the number of local windows of the images [31].

Fig. 6 shows the proposed experimental procedure (EP) diagram for fine-tuning and the evaluation metric conducted in this work. After classifying Gmapping initialization parameters, every group of parameters is subjected to the EP separately to assess how the configuration would affect that group. After collecting the best configurations at one configuration, the results will be reached. The experimental procedure execution is repeated five times for every best configuration due to random internal terms in the algorithm. The final MSSIM value is obtained by taking the mean of those five values.

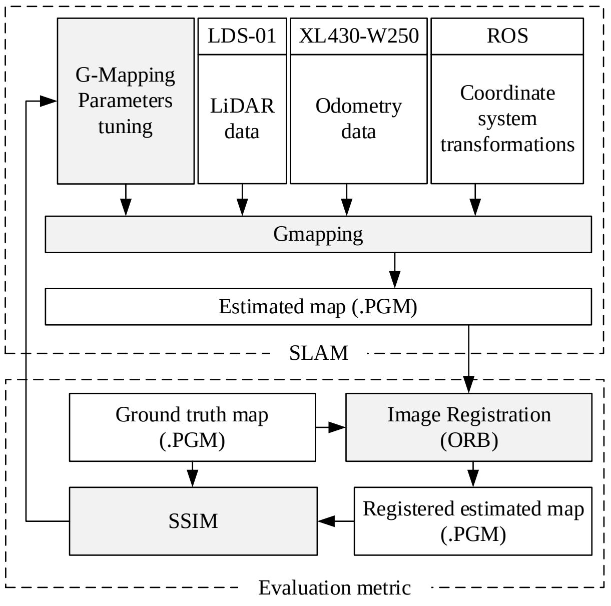


Figure. 6 Proposed EP diagram

5. Results and discussions

The study evaluates the accuracy of estimated maps for a specific robot model and an indoor environment, considering the impact of Gmapping initialization parameters on hardware resource utilization and map correction estimation. As Gmapping relies on probabilistic methods, accurate tuning of noise assumptions and initialization parameters is critical to obtaining high-quality results. In order to optimize the results, it is necessary to evaluate the effectiveness of each parameter group by tuning their initialization parameters. This process will facilitate the inspection of the results and determine the recommended parameters to be adjusted. Table 2 shows the default parameter configuration, and the parameters are tuned in a specific order, starting with OGM resolution parameters, followed by the measurement update frequency, motion model parameters, scan matching parameters, and particle filter parameters. The default configuration achieves an SSIM score of 0.78849 (78.849%), and fine-tuned parameters are categorized into six groups as follows:

Group 1: Coordinate system transformations parameters Gmapping uses three frames to predict the robot's position on the estimated map. The word frame represents the fixed one, and the odometry frame represents the drifting error between the mobile robot base frame and the fixed frame on the estimated map. The period for issuing frames transformations to generate the map while estimating the self-position is 0.05 seconds, given as a default value of the transform_publish_period parameter. For this parameter, it is not recommended to change the value of the publishing period since that will affect the whole synchronization process of the algorithm structure.

Group 2: OGM parameters

The default values of the initial map dimensions constrain the initial map dimensions, so they do not affect the mapping procedure. Otherwise, the delta parameter is the scale of one pixel in the map in meters per occupancy grid cell. In addition, the occ_thresh parameter is the threshold of each grid cell probability value. If the probability value is equal to or greater than the value of this parameter, it is judged to be an obstacle. After tuning those two parameters, the occupancy threshold parameter's value better remains as (0.25), but for more precise gids of the map, the delta is fine-tuned to the value of (0.025 m). that depends on the scale of the real environment to be represented as OGM.

Group 3: Measurement update parameters

Repeated measurements in the same location can skew the results, making it seem more likely that the particles are not what they seem. Therefore, the subsequent assessment is delayed until a predetermined amount of travel is elapsed. Those parameters represent how much a robot must translate or rotate to process the latest scan data. Only after the robot reaches those thresholds do a new measurement. Even if the robot is not moving during the preceding linear and angular updates, observation will be conducted after a conditional time. If temporalUpdate is set to minus, there will be no time-based observations. There is a trade-off between accuracy and computation in selecting these parameters. Setting an enormous value for these two parameters will reduce the computation but would result in poor map quality. After several tuning tests, the methodology of fine-tuning the motion threshold parameters is to select values more extensive than the maximum liner or angular movement done per one second (0.3 m or 1 rad, respectively). Moreover, the temporal threshold's fine-tuned values range between 1 to 3 seconds.

Group 4: Motion model parameters setting the odometrical error parameters to zeros means the odometry data is ideal. The four parameters represent pure translational (srr), translational to rotational (srt), rotational to translational (str), and pure rotational (stt) noise components. Fig. 7 shows the simulation of tuning different values of motion model parameters. After many considerable empirical shuffling of the motion model parameters, the fine-tuned variances are: 0.05, 0.10, 0.09, and

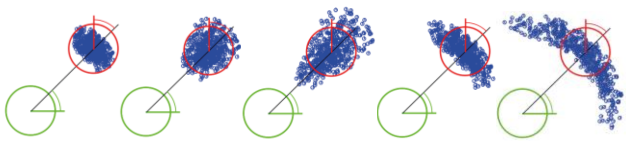


Figure. 7 Odometry sample model: (a) all parameters have the same prediction value, (b) the impact of srr, (c) the impact of srt, (d) the impact of str, and (e) the impact of stt

0.15 for s_{rr}, s_{rt}, s_{tr}, and s_{tt} respectively. However, even if the tuning of error values is relatively over-estimated, a particle filter can solve that as a Bayes filter by highly sparse particles. The effectiveness of tuning this group is indicated in Fig. 7. It significantly impacts localization accuracy and, implicitly, mapping accuracy. Motion model noise parameters are fine-tuned due to empirical shuffling. However, even if the tuning of error values is relatively over-estimated, a particle filter can solve that as a Bayes filter by highly sparse particles.

Group 5: Scan matching and LiDAR parameters. The maxUrange parameter was set to be greater than the maximum characterized range of the LiDAR (MaxR-LiDAR) sensor to depict open areas as empty on the map. In addition, the maxRange parameter was set to be less than or equal to MaxR-LiDAR. However, it was determined that it is more efficient to shorten the length of the usable rays, as errors tend to be larger at longer distances. The parameters utilized in this work were based on the specifications of the LDS-01 LiDAR sensor (MaxR-LiDAR = 3.5 m). During the mapping process, it was observed that specific cells were classified as occupied, despite being outside the room outline and vice versa. This issue was addressed by setting these parameters as maxRange = maxURange < MaxR-LiDAR.

The Throttle_scans parameter was also investigated, which denotes the number of un-processed scans. If a value of three is set, indicating that one scan out of every three was not processed. However, it is crucial to note that using the default value may be necessary as certain essential features could be impacted. To minimize computational processing, setting the value to one is recommended.

In addition, the standard deviation of the score function (sigma) and the standard deviation of the likelihood function (lsigma) of a single beam of the LiDAR was evaluated. The optimal values for these parameters were found to be the default values. The kernelSize parameter was kept at value one as the map discretizes into squared grids, and the search space in the optimization step of scan matching represents the nine grids surrounding the targeted grid.

The initial linear or angular step size of the scan matcher optimization (lstep and astep) was recommended to be the same as the delta parameter, representing the grid size. Additionally, two other refinement steps increased the iterations parameter value to obtain a more optimized pose. To further optimize the process, the lskip parameter was tuned, and it was found that skipping one or two beams in each scan is a viable option, as not all laser data is always necessary.

Likelihood sampling parameters concerning the scan matcher's calculation of the likelihood field are the default settings for a grid map of 5 cm. In this study, the map resolution was set as 2.5 cm. By adjusting the sampling values, the best choices are indicated in Table 3. The grid dimension in the estimated map was fine-tuned to be 0.025 m, and the scan matching parameters, such as lstep, astep, and sampling likelihood parameters, were adjusted accordingly.

Group 6: PF parameters

As a PF approach, selecting an appropriate number of particles is crucial to balancing computational efficiency and accuracy. However, it is essential to note that increasing the particle count can lead to an increased computational burden. As such, in this work, the tuning of particle number was approached as a last resort, and a small particle population of up to 20 particles was found to be suitable for obtaining high-performance and accurate maps. It is also worth noting that the particle number and resampling threshold are interdependent PF parameters. A high particle number can slow down the processing of the entire simultaneous localization and mapping (SLAM) system. Therefore, the resampling threshold is advised to be set at a ratio of half the particle number.

The map_update_interval parameter, which denotes the frequency at which map recalculations are performed, was set to 5 seconds in this study. It is important to note that a lower value for this parameter can increase the computational load on the system, while a higher value can improve map accuracy, depending on the environmental features. Additionally, the minimumScore parameter serves as a threshold for considering the outcome of the scan matching to be good. It is, and its default value is set at 0. A high value for this parameter can lead to a higher reliance on odometry alone, effectively turning Gmapping into a mapping algorithm only. The ogain parameter was found to be best set at its default value after several tuning experiments. Similarly, the resampling threshold parameter was found to be best set at its default value after several shuffling tests.

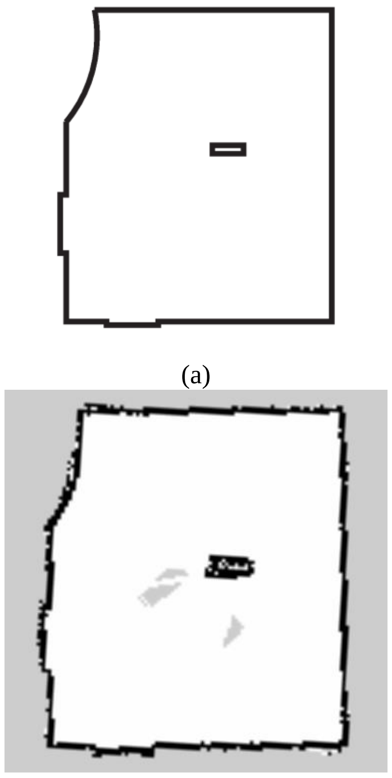
Table 3. The optimal configuration

Parameter	**Range**		**Optimal Value**
	Min.	**Max.**	
Group (2): OGM parameters			
xmin	-30	-20	-25
ymin	-30	-20	-25
xmax	-30	-20	-25
ymax	-30	-20	-25
delta	0.025	0.05	0.025
occ_thresh	0.25	0.25	0.25
Group (3): Measurement Update Parameters			
linearUpdate	0.2	0.5	0.3
angularUpdate	0.5	1	1
temporalUpdate	1	4	2
Group (4): Motion model parameters			
srr	0.001	0.05	0.05
str	0.001	0.1	0.1
str	0.02	0.2	0.09
stt	0.1	0.2	0.15
Group (5.1): Scan matching and LiDAR parameters			
maxRange	3.94	3.94	3.49
maxUrange	3.4	3.49	3.49
throttle_scans	1	1	1
sigma	0.05	0.5	0.05
lsigma	0.075	0.075	0.075
kernelSize	1	1	1
lstep	0.025	0.05	0.025
astep	0.025	0.05	0.025
iterations	5	7	7
lskip	0	0	0
Group (5.2): Likelihood sampling parameters			
llsamplerange	0.005	0.01	0.005
llsamplestep	0.005	0.01	0.005
lasamplerange	0.0025	0.005	0.0025
lasamplestep	0.0025	0.005	0.0025
Group (6): PF parameters			
particles	20	75	25
map_update_interval	3	5	5
minimumScore	0	50	0
ogain	3	3	3
resampleThreshold	0.5	0.75	0.5

It was found that the impact of fine-tuning initialization parameters on the accuracy of the mapping algorithm was significant. Table 3 presents the best range of values for all 32 parameters and the optimal value chosen in the final test to achieve the best accuracy for this study's specific environmental and robot model.

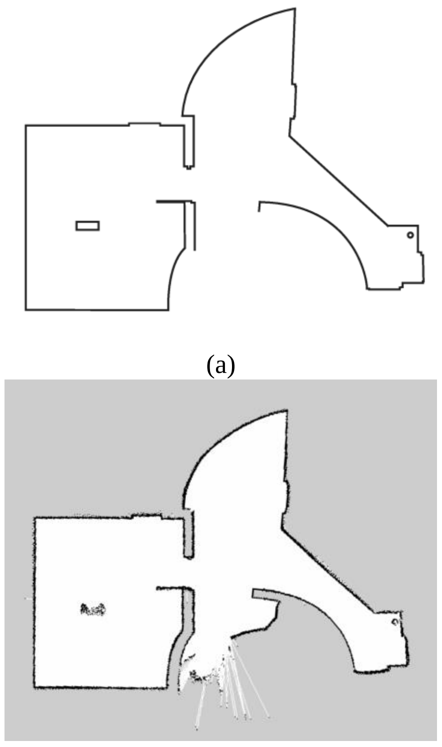
Fig. 8 illustrates the visual representation of the accuracy achieved in 1st Exp. utilizing the default configuration without any tuning applied. The

Received: January 15, 2023. Revised: February 27, 2023.


(a)

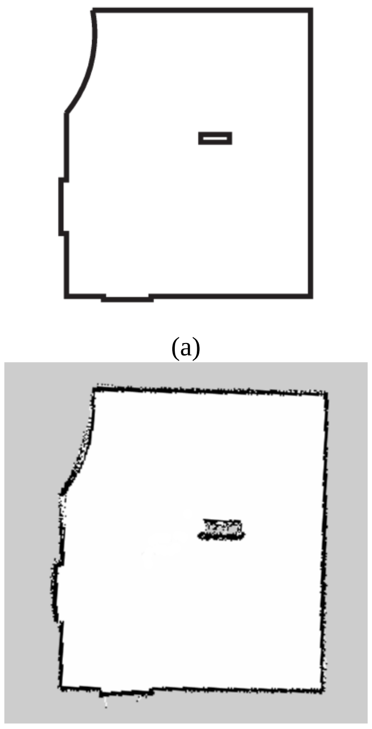
(b)

Figure. 8 A visual inspection of the 1st Exp. environment for the difference between: (a) its ground truth and (b) the estimated OGM via default configuration


(a)

(b)

Figure. 10 A visual validation inspection of the 2nd Exp. environment for the difference between: (a) its ground truth and (b) the estimated OGM via the final optimal configuration


(a)

(b)

Figure. 9 A visual inspection of the 1st Exp. environment for the difference between: (a) its ground truth and (b) the estimated OGM via the final optimal configuration

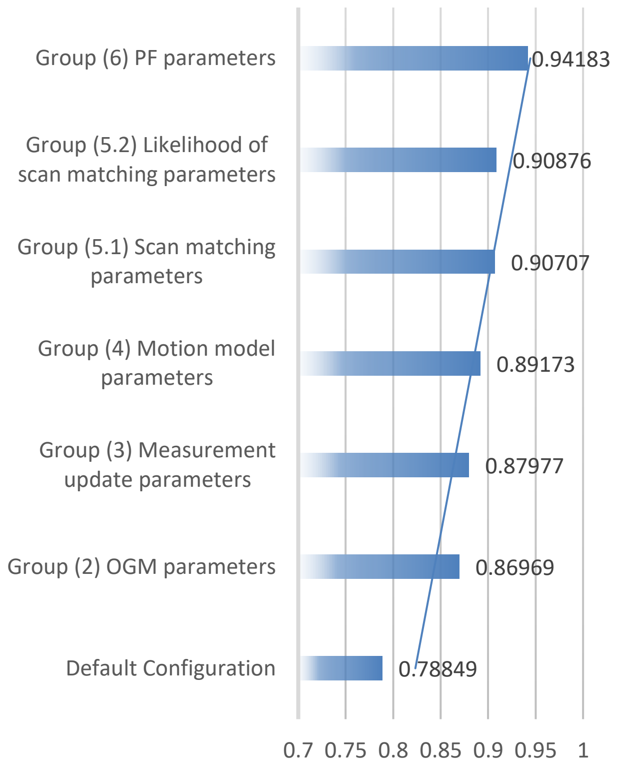


Figure. 11 MSSIMs of fine-tuning every group of parameters for 1st Exp

straight and curvature outlines appear noisy, and areas in the environment remain grey, indicating unknown areas. Additionally, Fig. 9 provides a visual inspection of the accuracy of the 1st experiment, which is nearly accurate. The straight curvature and obstacle outlines are clear, and no areas within the targeted environment remain undiscovered. Finally, the accuracy of the estimated OGM based on the optimal fine-tuned configuration has been tested and validated in the 2nd Exp., which is illustrated as a visual inspection in Fig. 10.

Each optimal configuration was repeated five times to obtain the mean of the results since the algorithm contains probabilistic and internal random terms. Upon comparing the accuracy of the default configuration at the bottom of Fig. 11 to the MSSIM values obtained after fine-tuning the four numerical groups of categorized parameters, it was observed that the MSSIM values increased from approximately ~78% to ~94%.

6. Conclusion

The results of this study show that Gmapping can effectively solve the localization and mapping problem in a room-scale environment with uncertain locomotion and provide a low-cost solution for optimizing the algorithm through parameters fine-tuning and a benchmark experimental procedure for evaluating the accuracy of the estimated map.

The highly accurate estimated map means implicitly highly accurate localization of the robot. Gmapping has 32 numerical initialization parameters that highly affect the accuracy of the resulting map depending on the environment scale, the hardware used, and locomotion characteristics. Setting significant parameters for Gmapping SLAM is unnecessary and can increase the hardware cost without improving the mapping results.

The parameters have been divided into categories based on the functioning of each parameter due to their large number. This work provides a reference for setting the parameters and Benchmark. Scan matching parameters such as *lstep*, *astep*, and sampling likelihood parameters are tuned according to the value of the delta parameter, which is tuned to be 0.025 m. Motion model noise parameters are fine-tuned due to empirical shuffling. However, even if the tuning of error values is relatively over-estimated, a particle filter can solve that as a Bayes filter by highly sparse particles. In this work, the effectiveness of Gmapping in solving the problem of particle depletion in RBPF is demonstrated by reducing the particle number below the default value due to the systemic tuning of the motion model parameters. The more fine-tuning of motion model parameters, the less requirement to increase the particle number, which impacts computational resources.

However, there are still many opportunities for future research that could further improve the performance of Gmapping in indoor environments. For example, additional sensors such as cameras or depth cameras could be used to provide additional information for the mapping and localization process, and machine learning techniques could be explored as a means of optimizing the selection and tuning of Gmapping parameters based on the characteristics of the environment and the hardware being used. Additionally, the performance of Gmapping could be evaluated in larger-scale or more complex environments or compared with other SLAM algorithms.

In conclusion, fine-tuning the Gmapping algorithm’s initialization parameters has significantly improved the accuracy of the generated 2D occupancy grid maps. By systematically classifying and optimizing these parameters, map accuracy is improved from a default configuration accuracy of 78.84% to a PF parameters accuracy of 94.18%, using a TB3B and a structured evaluation procedure incorporating image registration and similarity measurement. This finding is essential for developing autonomous navigation systems that rely on accurate maps for localization and obstacle avoidance. It also highlights the importance of carefully considering the selection and optimization of algorithms and parameters to achieve optimal performance.

Conflicts of interest

The authors declare that they have no conflicts of interest. This research was conducted independently and without funding from any external sources.

Author contributions

Conceptualization: Zuhair A. Ahmed and Safanah M. Raafat; methodology: Zuhair A. Ahmed, Safanah M. Raafat; software: Zuhair A. Ahmed; validation: Zuhair A. Ahmed, Safanah M. Raafat; formal analysis: Zuhair A. Ahmed; investigation: Zuhair A. Ahmed and Safanah M. Raafat; resources: Zuhair A. Ahmed and Safanah M. Raafat; data curation: Zuhair A. Ahmed; writing-original draft preparation: Zuhair A. Ahmed, Safanah M. Raafat; writing-review and editing: Zuhair A. Ahmed and Safanah M. Raafat; visualization: Zuhair A. Ahmed; supervision: Safanah M. Raafat; project

administration: Safanah M. Raafat and Zuhair A. Ahmed; funding acquisition, Not funded.

Acknowledgments

The authors acknowledge Mr. Riadh Dhaoui's contribution in providing open-source SSIM and image registration code on Github, which served as a valuable starting point for the analysis.

References

[1] M. Abed, O. Lutfy, and Q. A. Doori, "A Review on Path Planning Algorithms for Mobile Robots", *Engineering and Technology Journal*, Vol. 39, No. 5A, pp. 804–820, 2021.

[2] B. K. Oleiwi, A. Mahfuz, and H. Roth, "Application of Fuzzy Logic for Collision Avoidance of Mobile Robots in Dynamic-Indoor Environments", In: *Proc. of International Conference on Robotics, Electrical and Signal Processing Techniques*, pp. 131–136, 2021.

[3] H. Hakim, Z. M. Alhakeem, and A. Fadhil, "Asus Xtion Pro Camera Performance in Constructing a 2D Map Using Hector SLAM Method", *Iraqi Journal of Computer, Communication, Control and System Engineering*, Vol. 21, No. 3, pp. 1–11, 2021.

[4] X. Liu, Y. Lin, H. Huang, and M. Qiu, "Comparative Analysis of Three Kinds of Laser SLAM Algorithms", *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 12453 LNCS, pp. 463–476, 2020.

[5] C. Kolhatkar and K. Wagle, "Review of SLAM Algorithms for Indoor Mobile Robot with LIDAR and RGB-D Camera Technology", In: *Innovations in Electrical and Electronic Engineering*, pp. 397–409, 2021.

[6] H. M. Hasan and T. H. Mohammed, "Implementation of Mobile Robot's Navigation using SLAM based on Cloud Computing", *Engineering and Technology Journal*, Vol. 35, No. 6, pp. 634–639, 2017.

[7] Y. J. Choi, I. N. A. Ramatryana, and S. Y. Shin, "Cellular Communication-Based Autonomous UAV Navigation with Obstacle Avoidance for Unknown Indoor Environments", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 2, pp. 344–352, 2021, doi: 10.22266/ijies2021.0430.31.

[8] K. Trejos, L. Rincón, M. Bolaños, J. Fallas, and L. Marín, "2D SLAM Algorithms Characterization, Calibration, and Comparison Considering Pose Error, Map Accuracy as Well as CPU and Memory Usage," *Sensors, Vol. 22, Page 6903*, Vol. 22, No. 18, p. 6903, 2022.

[9] W. G. Teame, Y. Yu, and W. Zhongmin, "Optimization of SLAM Gmapping based on Simulation", *International Journal of Engineering Research and*, Vol. 9, No. 4, pp. 74–81, 2020.

[10] H. Wang, M. Huang, and D. Wu, "A Quantitative Analysis on Gmapping Algorithm Parameters Based on Lidar in Small Area Environment", *Lecture Notes in Electrical Engineering*, Vol. 594, pp. 480–492, 2020.

[11] I. A. Putra and P. Prajitno, "Parameter Tuning of G-mapping SLAM (Simultaneous Localization and Mapping) on Mobile Robot with Laser-Range Finder 360° Sensor", In: *Proc. of 2019 2nd International Seminar on Research of Information Technology and Intelligent Systems, ISRITI 2019*, pp. 148–153, 2019.

[12] W. A. S. Norzam, H. F. Hawari, and K. Kamarudin, "Analysis of Mobile Robot Indoor Mapping using GMapping Based SLAM with Different Parameter", *IOP Conference Series: Materials Science and Engineering*, Vol. 705, No. 1, 2019.

[13] Y. Abdelrasoul, A. B. S. H. Saman, and P. Sebastian, "A quantitative study of tuning ROS gmapping parameters and their effect on performing indoor 2D SLAM", In: *Proc. of 2016 2nd IEEE International Symposium on Robotics and Manufacturing Automation, ROMA 2016*, 2017.

[14] Z. Liu, Z. Cui, Y. Li, and W. Wang, "Parameter optimization analysis of gmapping algorithm based on improved RBPF particle filter", *Journal of Physics: Conference Series*, Vol. 1646, No. 1, 2020.

[15] R. Latif, K. Dahmane, and A. Saddik, "SLAM Algorithm: Overview and Evaluation in a Heterogeneous System", In: *Proc. of Enabling Machine Learning Applications in Data Science*, Singapore, pp. 165–177, 2021.

[16] R. Dhaoui and A. Rahmouni, "Mobile Robot Navigation in Indoor Environments: Comparison of Lidar-Based 2D SLAM Algorithms", *Lecture Notes in Mechanical Engineering*, pp. 569–580, 2022.

[17] X. Fan, Y. Wang, and Z. Zhang, "An evaluation of Lidar-based 2D SLAM techniques with an exploration mode", *Journal of Physics: Conference Series*, Vol. 1905, No. 1, p. 012021, 2021.

[18] X. Fan, Y. Wang, and Z. Zhang, "An

evaluation of Lidar-based 2D SLAM techniques with an exploration mode", *Journal of Physics: Conference Series*, Vol. 1905, No. 1, 2021.

[19] Q. B. Zhang, P. Wang, and Z. H. Chen, "An improved particle filter for mobile robot localization based on particle swarm optimization", *Expert Systems with Applications*, Vol. 135, pp. 181–193, 2019.

[20] H. Taheri and Z. C. Xia, "SLAM; definition and evolution", *Engineering Applications of Artificial Intelligence*, Vol. 97, p. 104032, 2021.

[21] D. Kakuma, S. Tsuichihara, G. A. G. Ricardez, J. Takamatsu, and T. Ogasawara, "Alignment of Occupancy Grid and Floor Maps Using Graph Matching", In: *Proc. of IEEE 11th International Conference on Semantic Computing, ICSC 2017*, pp. 57–60, 2017.

[22] P. Sankalprajan, T. Sharma, H. D. Perur, and P. S. Pagala, "Comparative analysis of ROS based 2D and 3D SLAM algorithms for autonomous ground vehicles", In: *Proc. of 2020 International Conference for Emerging Technology, INCET 2020*, 2020.

[23] B. K. Oleiwi, R. A. Jarrah, H. Roth, and B. I. Kazem, "Integrated motion planning and control for multi objectives optimization and multi robots navigation", *IFAC-PapersOnLine*, Vol. 28, No. 10, pp. 99–104, 2015.

[24] S. Oajsalee, S. Tantrairatn, and S. Khaengkarn, "Study of ROS Based Localization and Mapping for Closed Area Survey", In: *Proc. of 2019 IEEE 5th International Conference on Mechatronics System and Robots, ICMSR 2019*, pp. 24–28, 2019.

[25] B. K. Oleiwi, "Scouting and controlling for mobile robot based raspberry Pi 3", *Journal of Computational and Theoretical Nanoscience*, Vol. 16, No. 1, pp. 79–83, 2019.

[26] Y. K. Tee and Y. C. Han, "Lidar-Based 2D SLAM for Mobile Robot in an Indoor Environment: A Review", In: *Proc. of 2021 International Conference on Green Energy, Computing and Sustainable Technology, GECOST 2021*, 2021.

[27] E. A. Luengo, "Gamma Pseudo Random Number Generators", *ACM Computing Surveys*, Vol. 55, No. 4, 2022.

[28] K. Sugiura and H. Matsutani, "An FPGA Acceleration and Optimization Techniques for 2D LiDAR SLAM Algorithm", *IEICE Transactions on Information and Systems*, Vol. E104.D, No. 6, pp. 789–800, 2021.

[29] Z. Su, J. Zhou, J. Dai, and Y. Zhu, "Optimization Design and Experimental Study of Gmapping Algorithm", In: *Proc of the 32nd Chinese Control and Decision Conference, CCDC 2020*, pp. 4894–4898, 2020.

[30] J. Ma, X. Jiang, A. Fan, J. Jiang, and J. Yan, "Image Matching from Handcrafted to Deep Features: A Survey", *International Journal of Computer Vision*, Vol. 129, No. 1, pp. 23–79, 2021.

[31] R. S. Jebur, C. S. Der, and D. A. Hammood, "A Review and Taxonomy of Image Denoising Techniques", In: *Proc. of 6th International Conference on Interactive Digital Media, ICIDM 2020*, pp. 1–6, 2020.