Bulletin of Electrical Engineering and Informatics
Vol. 7, No. 3, September 2018, pp. 433~441
ISSN: 2302-9285, DOI: 10.11591/eei.v7i3.1275

GA-based Optimisation of a LiDAR Feedback Autonomous Mobile Robot Navigation System

Siti Nurhafizah Anual[1], Mohd Faisal Ibrahim[2], Nurhana Ibrahim[3], Aini Hussain[4], Mohd Marzuki Mustafa[5], Aqilah Baseri Huddin[6], Fazida Hanim Hashim[7]
[1,2,4,5,6,7]Centre for Integrated Systems Engineering and Advanced Technologies (INTEGRA), Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia
[2,3,4,5,6,7]Electrical and Electronic Engineering Programme (PKE), Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia

Article Info

Article history:

Received May 21, 2018
Revised Jul 31, 2018
Accepted Aug 10, 2018

Keywords:

Autonomous robot navigation
GA optimization
Path planning

ABSTRACT

Autonomous mobile robots require an efficient navigation system in order to navigate from one location to another location fast and safe without hitting static or dynamic obstacles. A light-detection-and-ranging (LiDAR) based autonomous robot navigation is a multi-component navigation system consists of various parameters to be configured. With such structure and sometimes involving conflicting parameters, the process of determining the best configuration for the system is a non-trivial task. This work presents an optimisation method using Genetic algorithm (GA) to configure such navigation system with tuned parameters automatically. The proposed method can optimise parameters of a few components in a navigation system concurrently. The representation of chromosome and fitness function of GA for this specific robotic problem are discussed. The experimental results from simulation and real hardware show that the optimised navigation system outperforms a manually-tuned navigation system of an indoor mobile robot in terms of navigation time.

Corresponding Author:

Mohd Faisal Ibrahim,
Centre for Integrated Systems Engineering and Advanced Technologies (INTEGRA),
Universiti Kebangsaan Malaysia,
43600 Bangi, Selangor, Malaysia
Email: faisal.ibrahim@ukm.edu.my

1. INTRODUCTION

Autonomous mobile robot navigation systems are a class of robot control systems in which the degree of autonomy is complex. The systems allow a robot to move from a location to another location without the intervention of human input or tele-operation. Autonomous mobile robots have wide applications ranging from domestic vacuum cleaners to self-driving cars. With the advancement of sensor and processor technologies, the implementation of light-detection-and-ranging (LiDAR) sensor into a robot navigation system has a significant impact to the way robots perceive its surrounding environments. In particular, input data from a LiDAR sensor change the behavior of robot control from reactive approach to deliberative approach.

In past few decades, there are various algorithms that have been developed to optimise robot navigation systems. The algorithms can be categorised into two categories i.e. 1) greedy algorithms, and 2) artificial intelligence (AI) techniques.

The greedy algorithms are a collection of methods that solve an optimisation problem by considering locally optimal choice at each single process. For example, a work by [1] find the shortest path by using graph-based algorithm for an indoor application to help visually impaired people to walk. The major feature of the algorithm is the minimisation of a number of turns in a travel path that helps visually impaired

people to walk easily. [2] implements A* algorithm to find near optimal path for a robot by reducing the processing time but sub-optimal in path length. Although such techniques can be considered as having fast processing time, the solution is always local optima.

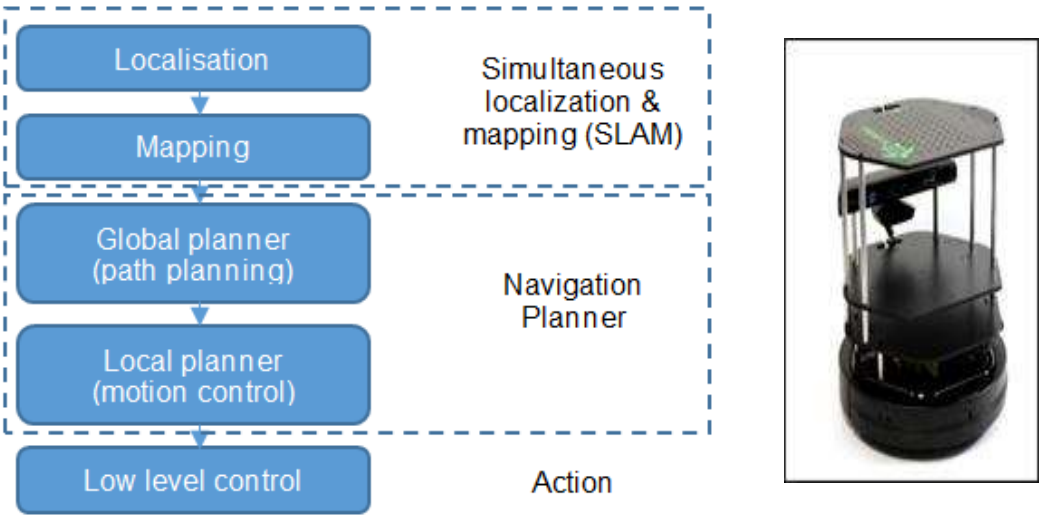
On another hand, there are wide spectrum of research works that use AI techniques to optimise path planners such as Genetic algorithm (GA), Fuzzy logic and Neural network. AI techniques are promising as the methods provide an effective mechanism to find a global optima solution of an optimisation problem. In particular, GA is a good choice at finding solution for problems with large search space. GA is a stochastic universal search technique that imitates the principle of natural biological evolution [3]. Many works have used GA as a tool to solve various path planning problems by taking advantage of its strong optimisation ability [4]. [5] applies GA to generate an efficient path of coverage regions for a vacuum robot to clean all accessible areas in an indoor room environment. [6] also uses GA to solely optimise a global planner of a field robot on a low resolution grid map. The GA is used to find the shortest distance for global path planning with random number of obstacles appeared in a map. [7] implements GA to generate fast evacuation plan to suitable exit doors based on the entered facility configuration. Meanwhile, [8] used a new population-based optimisation algorithm called Wind Driven Optimization (WDO) to optimise the fuzzy logic-based path planning of a mobile robot in unknown static and dynamic environments. Although these works show the applicability of GA in various navigation configurations, the optimisation was done in isolation from one part of the navigation system to another part.

This work focuses on implementing GA to tune both the local and global planners of a robotic navigation system concurrently. Both components have a list of parameters that determines the selection of paths and trajectories of the robot which reflect the most on the robot navigation performance. Setting up all parameters with appropriate values is non-trivial since there is no clear relationship between parameters that yield the most optimum configuration. Thus, a heuristic approach by using GA is proposed to automatically tune necessary parameters of the navigation system. GA is chosen since the searching process can be distributed effectively for a large dimension of search space of the system.

The rest of this paper is organised as follows. In section 2, we describe the details of research method used in this work. Section 3 discusses the result of the proposed method based on GA. Finally, we concludes this work in section 4.

2. RESEARCH METHOD

A deliberative robot navigation system is a complex system with various parameters that might conflict with each other. The deliberative navigation system used in this work is shown in Figure 1(a). Figure 1(b) displays the physical image of a robot (Turtlebot2) equipped with an imitated LiDAR sensor (Kinect) being studied.



(a) (b)

Figure 1. (a)A robot deliberative navigation system (b)Turtlebot2 with Kinect camera

Localisation, mapping, global planner and local planner are the main components in a robot deliberative navigation system [9]-[13]. Localisation provides the mechanism of determining the current location of the robot. Mapping is the process of building and maintaining the model of an environment explored by the robot. Both components are usually tie together as mapping requires the robot location as the input and localisation needs a current map to localise the robot. On another hand, global planner and local planner are two important components that determine the path to be taken by the robot to reach its destination. Global planner is responsible to find the shortest and safest path connecting the robot current location to an assigned destination location taking into account the current map and static obstacles. Meanwhile, local planner is used to decide the trajectory to be taken by the robot by considering the suggested global path, distance to the destination point and dynamic obstacles. Local planner is usually run at higher frequency than global planner. Finally, the trajectory information is sent to the low level control to translate it into the rotation of robot wheels.

The navigation system with improper parameters setting may cause the robot to take longer time to reach its destination. Due to this fact, a proper method to identify the best parameters of the system must be developed to achieve optimal navigation performance. An optimal robot navigation system allows the robot to perform efficiently without any collision with static and dynamic obstacles. In this work, parameters in the local planner and global planner of the robot navigation system are formulated in the form of a chromosome to be fed into a GA program. The process of optimising the parameters of local and global planners of the navigation system using GA has three stages:

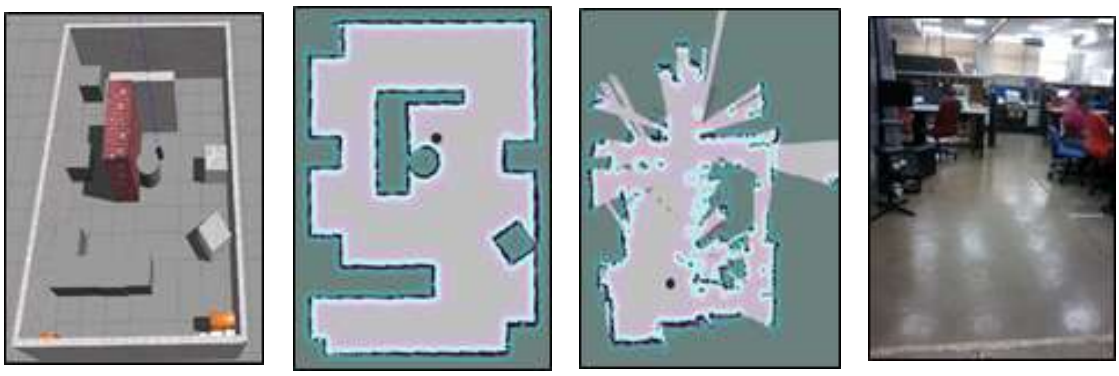
a. Acquire samples of environments.
b. Optimise parameters using GA.
c. Validate the output from GA.

All three stages are discussed in the next sub-sections.

2.1. Acquire Samples of Environments

Samples of environments where the robot will perform the navigation task is required as the input to our proposed GA-based optimisation for a robot navigation system. The samples can be collected in the form of an occupancy grid map of environments representing free space and obstacles location on the map.

In this work, we acquired a map of an environment by moving the robot around with tele-operation procedures. Two maps are gathered: i) a map of a simulated environment created using Gazebo 7 software [14], and ii) a map of a real world environment in our Control laboratory. The developed maps are used as the ground space to optimise and test the robot navigation system. Figure 2(a) and 2(b) show the resultant maps for the simulated environment and the real-world environment, respectively.



(a) (b)

Figure 2. (a) Simulated environment with its corresponding grid map, (b) Real world enviroment (our Control laboratory) with its corresponding grid map

2.2. Optimise parameters using GA

The process to develop a GA program to optimise the robot navigation system involves three main GA components to be configured:

a. Chromosome representation
b. Fitness function
c. GA operators

2.2.1. Chromosome Representation

In the first step, a chromosome or an array of binary numbers that defines a candidate solution is designed. Here, we set 7 main parameters that need to be optimised by GA for both local and global planners altogether. Note that, we adopt the framework given in an open source Robotic Operation System software (ROS) [15] i.e. move_base package to build the robot navigation system. The global planner is set with local-minimum free navigation function (NF1) while the local planner utilises Dynamic Window Approach (DWA) method [16]. For global planner, the parameters are:

a. old_navfn_behavior–to choose between flexible function or fixed function.
b. use_quadratic–to choose the calculation of potential field either quadratic or linear.
c. use_dijkstra–to choose between Djikstra algorithm or A* algorithm.
d. use_grid_path–to choose between grid boundaries follower or gradient descent method to create path.

For DWA local planner, the parameters involved are:

a. path_distance_bias–weight on the distance between robot and global path.
b. goal_distance_bias–weight on the distance between robot and goal position.
c. occdist_scale–weight on the distance between robot and the nearest obstacle.

These DWA parameters are used in the decision model of DWA as in Equation 1:

$$Trajectory\_score = path\_distance\_bias*d1 + goal\_distance\_bias*d2 + occdist\_scale*d3 \quad (1)$$

where d1 is the distance between robot and global path, d2 is the distance between robot and goal position, and d3 is the distance between robot and the nearest obstacle.

Each chromosome contains all above 7 parameters. Table 1 shows the range of value and number of bits for the selected parameters. Note that the standard binary to decimal conversion is applied in this case.

Table 1. Details of Selected Parameters

Parameter	Range	Number of bits
Global Planner		
old_navfn_behavior	0 or 1	1
use_quadratic	0 or 1	1
use_dijkstra	0 or 1	1
use_grid_path	0 or 1	1
Local Planner		
path_distance_bias	0 to 100	10
goal_distance_bias	0 to 100	10
occdist_scale	0 to 1	10

2.2.2. Fitness Function

Fitness function is used to evaluate each chromosome and its fitness value. The fitness value represents the performance of a chromosome in solving a specific given problem. The fitness function of the robot navigation system in this work is designed with an offline learning approach paradigm. The offline learning takes the advantage of using simulation tools to simulate the robot movement in a simulated environment with a specific set of navigation parameters rather than running the real robot in a real world environment. This is useful to reduce the time taken to evaluate all candidate solutions suggested by GA by speeding up the simulation speed. We measure the performance of robot navigation based on time it takes to reach a destination point from a starting point. Thus, the fitness function for the GA is shown in Equation 2.

$$Fitness = 1 - \frac{T_{arrive}}{T_{simulation}} \quad (2)$$

where, T_{arrive} is time taken by the robot to arrive at an assigned destination, and $T_{simulation}$ is the maximum allowable time to perform navigation.

Every time a new chromosome is required to be evaluated, binary values from the chromosome are converted into their corresponding decimal values and are transferred to appropriate planner parameters. Then a simulation run with a fixed starting point and a destination point on a simulated map as in Figure 2(a) is executed. At the end of the execution, time for the robot to arrive at the destination is recorded. Finally, fitness value is calculated. A chromosome with higher fitness is considered as having better navigation performance compared to a chromosome with lower fitness.

2.2.3. GA Operators

The complete process of GA in this work is summarised in a flowchart of Figure 3. The process starts by selecting the size of the generation, the population size and appropriate GA operators. After that, a population of chromosomes is initialized according to the above mentioned chromosome representation. Then, the fitness of every chromosome is evaluated by performing a simulated navigation run and subsequently using the abovementioned fitness function. Following that, we perform GA operators of selection, crossover and mutation in sequence. We set selection operator with Roulette Wheel method, crossover operator with one-point crossover (0.9 probability) and mutation operator with random bit mutation (0.01 probability). Table 2 shows the configuration of the GA parameters used in this work. The value for those parameters are set heuristically. For more details process of standard binary GA.

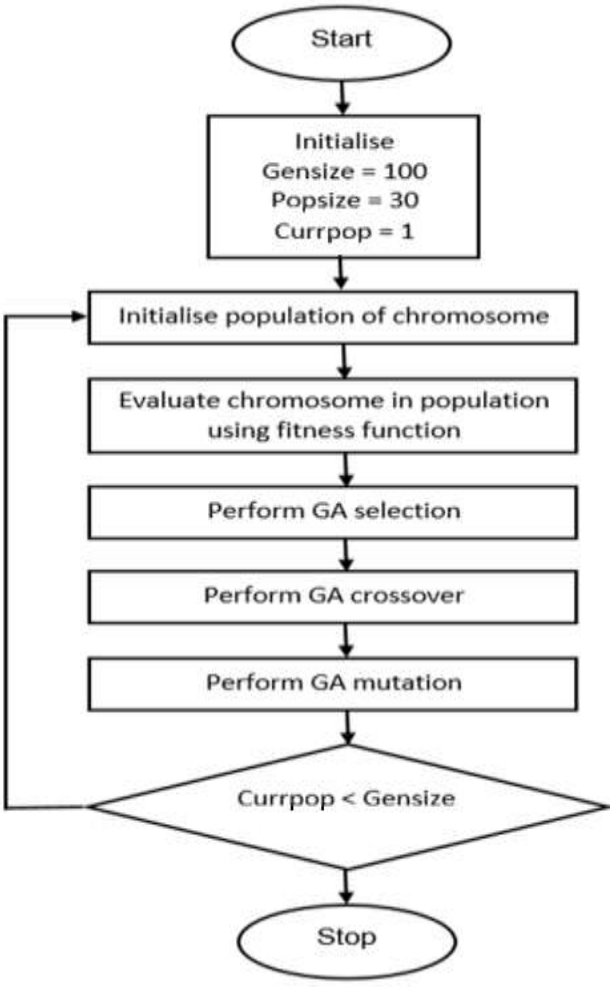


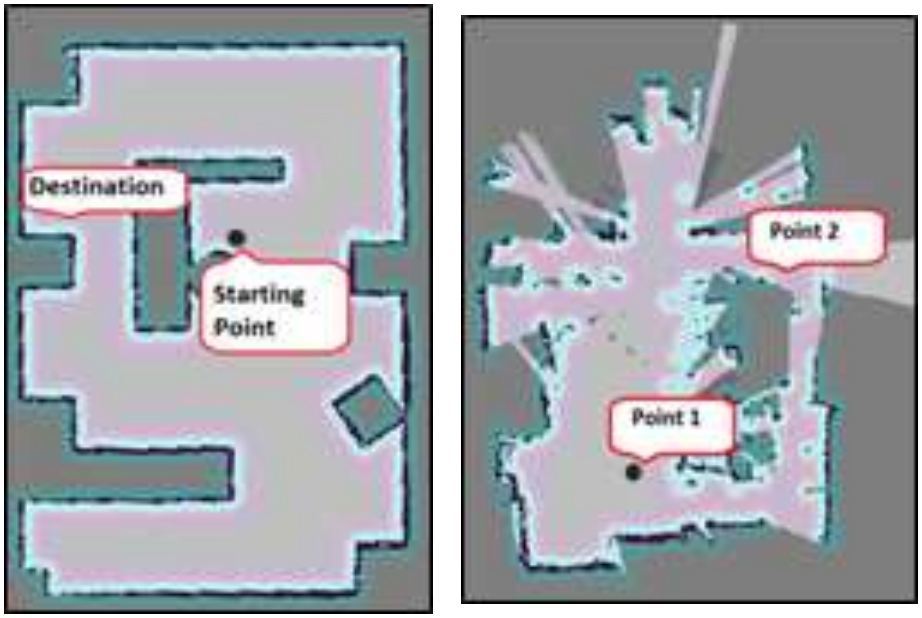
Figure 3. Flowchart of complete GA process

Table 2. GA Parameters used in This Work

Parameter	Value
No. of generation	100
Population size	30
Type of selection	Roulette wheel
Type of crossover	One-point
Probability of crossover	0.9
Type of mutation	Random bit
Probability of mutation	0.01

2.3. Validate the Output from GA

To validate the result from GA, the final configuration of navigation system with optimal parameters is tested on a real-world environment. For the benchmarking purpose, the performance comparison in terms of navigation time is made between the default and GA-tuned navigation systems. Again, for this task, the starting point and the destination point are set as in Figure 4.



(a) (b)

Figure 4. Assigned starting and destination points in (a) a simulated environment and (b) a real-world environment

3. RESULTS AND ANALYSIS

Based on the methodology explained in section 2, we carried out experiments to optimise the robot navigation system as follows. First, we ran GA process embedding a simulated environment as in Figure 2(a) to perform fitness evaluation. After a few generations run, the near-optimal navigational parameters were found. Based on these parameters setting, we verified the navigation system is optimal by assessing the navigation performance on the same simulated environment and a real world environment as in Figure 4 with 20 and 15 repetitive runs, respectively. The results are explained next.

3.1. GA Results

The optimal value for each parameter of local and global planners found by the proposed GA program is shown in Table 3. In comparison with the default manually tuned system, the results show that for global planner parameters, GA-tuned system has similar setting except for "use_djikstra" parameter where the GA-tuned system prefers A* algorithm rather than Djikstra algorithm. For local planner parameters, all three parameters have different values from the default system. The GA-tuned suggests that a higher weightage is given to "path_distance_bias" while weightage for "goal_distance_bias" and "occdist_scale" are maintained low.

Table 3. Parameters Setting for GA-tuned Navigation System and the Default Navigation System

Parameter	GA-tuned system	Default system
old_navfn_behavior	0 (false)	0 (false)
use_quadratic	1 (true)	1 (true)
use_dijkstra	0 (false)	1 (true)
use_grid_path	0 (false)	0 (false)
path_distance_bias	95.210	64.0
goal_distance_bias	6.354	24.0
occdist_scale	0.02151	0.5

3.2. Testing Results

To test the effectiveness of the modified parameter values by GA-tuned navigation, performance comparison between GA-tuned system and the default system is conducted. For the first testing, the same simulation run when executing the GA program is repeated for both systems. The time of simulation is set at 100 seconds. For statistical purpose, 20 simulation runs are executed for each system.

Table 4 shows the result of both systems on the simulated environment. From the table, it is found that in 6 out of 20 runs, the robot does not arrive at the assigned destination by using the default system. In contrast with GA-tuned system, the performance is improved where only 2 out of 20 simulation runs that the robot does not arrive at the assigned destination within 100 seconds.

Table 4. Time (in seconds) taken for the Robot to Navigate from a Starting Point to a Destination Point set in a Simulation Environment of Figure 4(a)

Run	Default system	GA-tuned system	Run	Default system	GA-tuned system
1	77	24	11	39	23
2	>100	42	12	>100	87
3	22	25	13	99	23
4	23	22	14	24	62
5	69	84	15	25	>100
6	89	96	16	>100	33
7	>100	77	17	22	82
8	>100	95	18	26	24
9	70	25	19	>100	>100
10	22	64	20	25	21

Based on the data from the Table 4, Figure 5 shows the box plot of the data for both systems. Median time taken for the robot to reach the destination by using the default system is 69.2 seconds while median time for GA-tuned system is 52 seconds. This value shows that median time improves for about 25% faster with GA-tuned system.

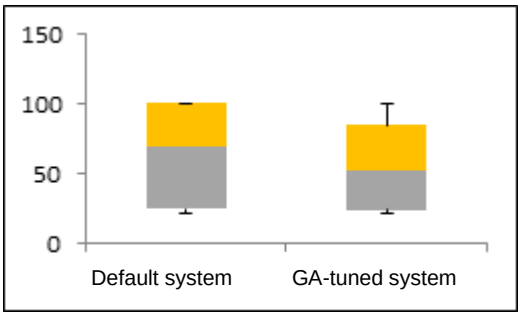


Figure 5. Box plot of time taken to reach destination point in a simulation environment for the default and GA-tuned system

In order to verify the performance in real hardware setting, both navigation systems are tested in our Control laboratory environment as in Figure 4(b). Table 5 and Figure 6 show the testing results of both systems in real world environment to navigate from Point 2 to Point 1. From the figure, it shows that median time for the default system is 25 seconds while media time for GA-tuned system is 21 seconds. The result shows that GA-tuned system outperforms the default system with 16% improvement.

Table 5. Time (in seconds) taken for the Robot to Navigate from Point 2 to Point 1 in a Real World Environment of Figure 4(b)

Run	Default system	GA-tuned system	Run	Default system	GA-tuned system
1	24	23	9	19	21
2	27	20	10	28	27
3	26	19	11	20	20
4	20	21	12	29	29
5	21	27	13	22	19
6	20	19	14	27	21
7	25	25	15	27	23
8	28	19			

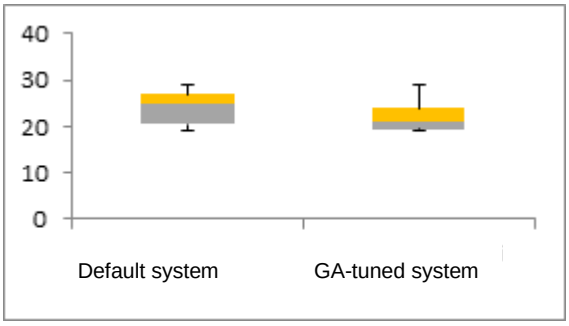


Figure 6. Box Plot of time taken to reach destination point in a real world environment for the default and GA-tuned system

The results from simulation and real world runs show that GA-tuned system produces better navigation time compared to the default system. Note that there is the degradation of improvement percentage in real world test compared to simulation test. However, the result is expected as this is known as "reality-gap" problem in evolutionary robotics field when learning is conducted with simulation environment (offline learning). One of the possible works to improve the problem is to implement combinatorial learning approach where offline learning with simulation and online learning with real hardware can be integrated together while performing GA process.

4. CONCLUSION

As a conclusion, the local and global planners of a robot navigation system are successfully optimised by using the proposed GA program. Based on the results, the improvement made by GA-tuned navigation system is 25% in simulation and 16% in real-world environment compared to the default navigation system. Thus, GA-tuned deliberative navigation system can improve the overall navigation performance of the robot by optimising its global and local planners.

ACKNOWLEDGMENT

This research was supported by the Ministry of Higher Education Malaysia and Universiti Kebangsaan Malaysia under the Fundamental Research Grant Scheme FRGS/1/2017/TK04/UKM/02/10.

REFERENCES

[1] Nandini D. and Seeja K. R., "A novel path planning algorithm for visually impaired people," *Journal of King Saud University - Computer and Information Sciences*, 2017.
[2] Guruji A. K., *et al.*, "Time-efficient A* Algorithm for Robot Path Planning," *Procedia Technology*, vol. 23, pp. 144–149, 2016.
[3] Aibinu A. M., *et al.*, "A novel Clustering based Genetic Algorithm for route optimization," *Engineering Science and Technology, an International Journal*, vol/issue: 19(4), pp. 2022–2034, 2016.
[4] Tuncer A. and Yildirim M., "Dynamic path planning of mobile robots with improved genetic algorithm," *Computers and Electrical Engineering*, vol/issue: 38(6), pp. 1564–1572, 2012.
[5] Yakoubi M. A. and Laskri M. T., "The path planning of cleaner robot for coverage region using Genetic Algorithms," *Journal of Innovation in Digital Ecosystems*, vol/issue: 3(1), pp. 37–43, 2016.

[6] Chaudhari Y. and Jena S., "Global Optimal Path Planning of Mobile Robot Using Genetic Algorithm," *International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering*, vol/issue: 4(5), pp. 493–497, 2017.
[7] Al Qhtani A. S., *et al.*, "A Fast Genetic Algorithm-based Evacuation Plan Generator," *Procedia Computer Science*, vol/issue: 109(2016), pp. 994–998, 2017.
[8] Pandey A. and Parhi D. R., "Optimum path planning of mobile robot in unknown static and dynamic environments using Fuzzy-Wind Driven Optimization algorithm," *Defence Technology*, vol/issue: 13(1), pp. 47–58, 2017.
[9] M. Plaza P., *et al.*, "Global and Local Path Planning Study in a ROS-Based Research Platform for Autonomous Vehiches," *Journal of Advance Transportation*, pp. 1-10, 2018.
[10] Darweesh H., *et al.*, "Open Source Integrated Planner for Autonomous Navigation in Highly Dynamic Environments," *Journal of Robotics and Mechatronic*, vol/issue: 29(4), pp. 1-6, 2017.
[11] Zhao C., *et al.*, "Building a grid-semantic map for the navigation of service robots through human-robot interaction," *Digital Communications and Networks*, pp. 253-266, 2015.
[12] Atyabi A. and Powers D. M. W., "Review of classical and heuristic-based navigation and path planning approaches," *International Journal of Advanced Computer Technology*, pp. 1-14, 2014.
[13] Fethi D., *et al.*, "Simultaneous localization, mapping, and path planning for unmanned vehicle using optimal control," *Advance in Mechanical Engineering*, vol/issue: 10(1), pp. 1-25, 2018.
[14] Nogueira L., "Comparative Analysis between Gazebo and V-REP Robotic Simulators," *Seminario Interno de Cognicao Artificial-SICA*, 2014.
[15] Peter C., "Integrating ROS and MATLAB," *IEEE Robotics and Automation Magazine*, pp. 18-20, 2015.
[16] Christian H., *et al.*, "Energy Efficient Dynamic Window Approach for Local Path Planning in Mobile Service Robotics," *IFAC-PapersOnLine*, vol/issue: 49(15), pp. 32-37, 2016.